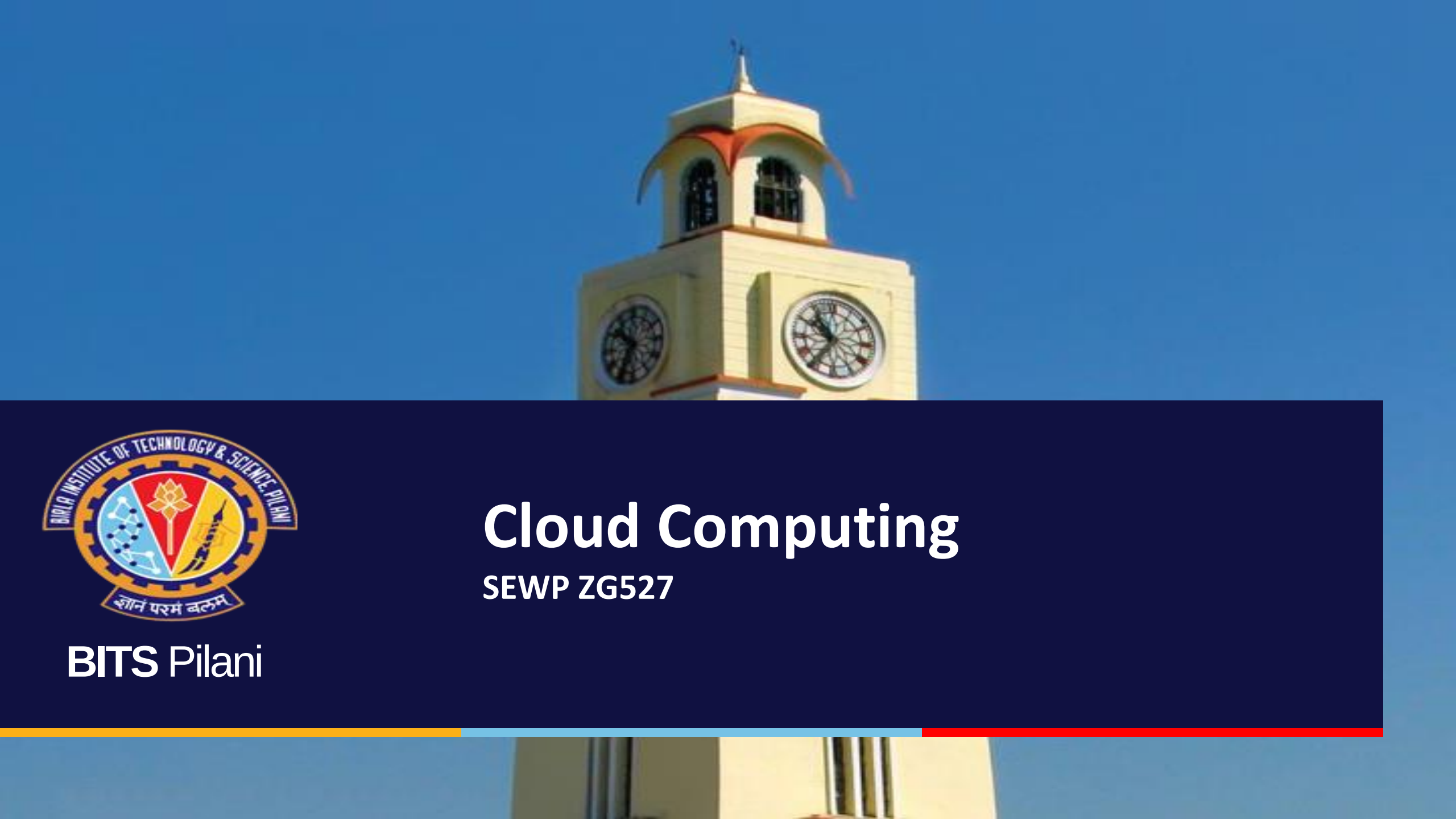
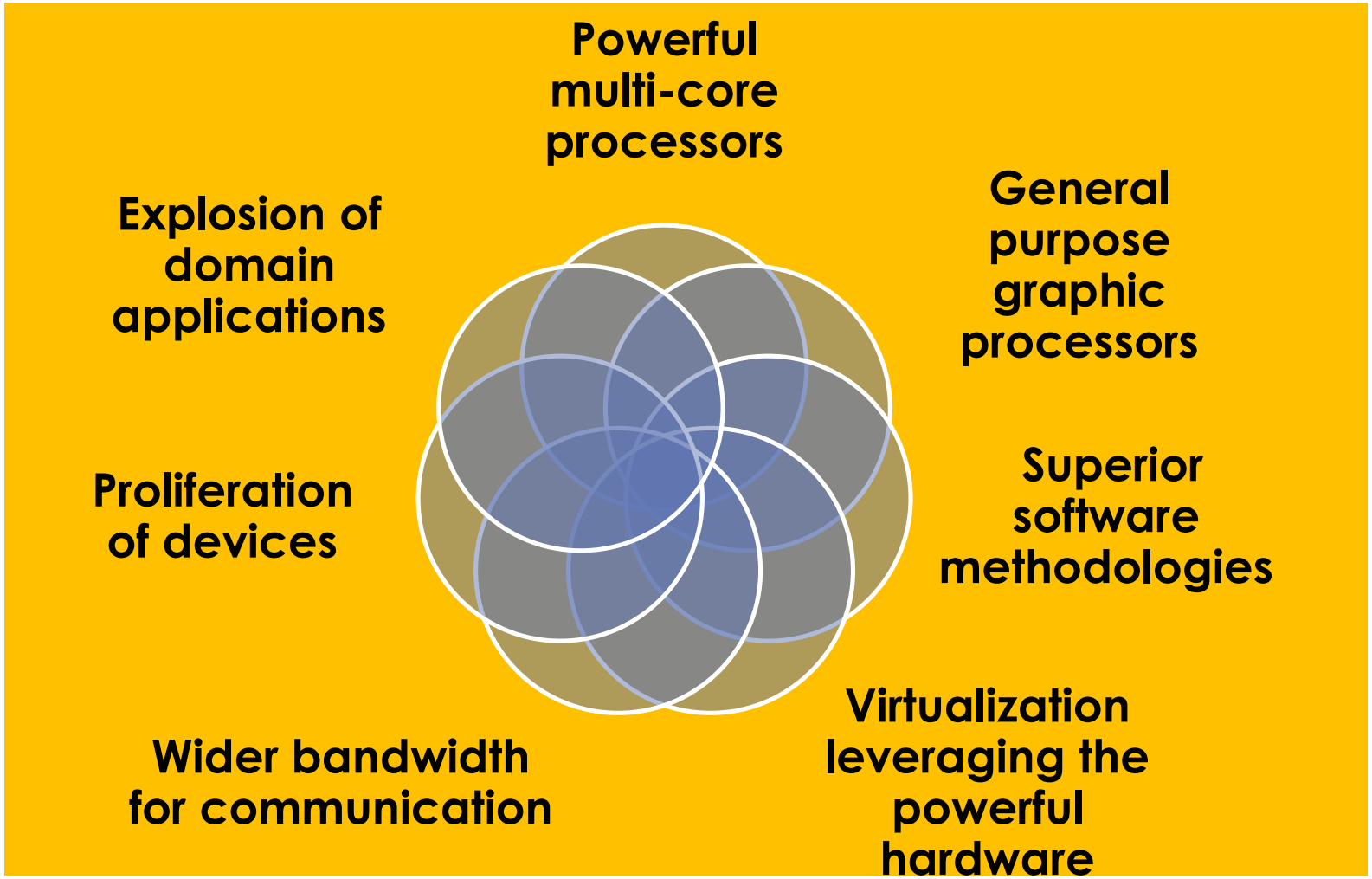


BITS Pilani

Cloud Computing

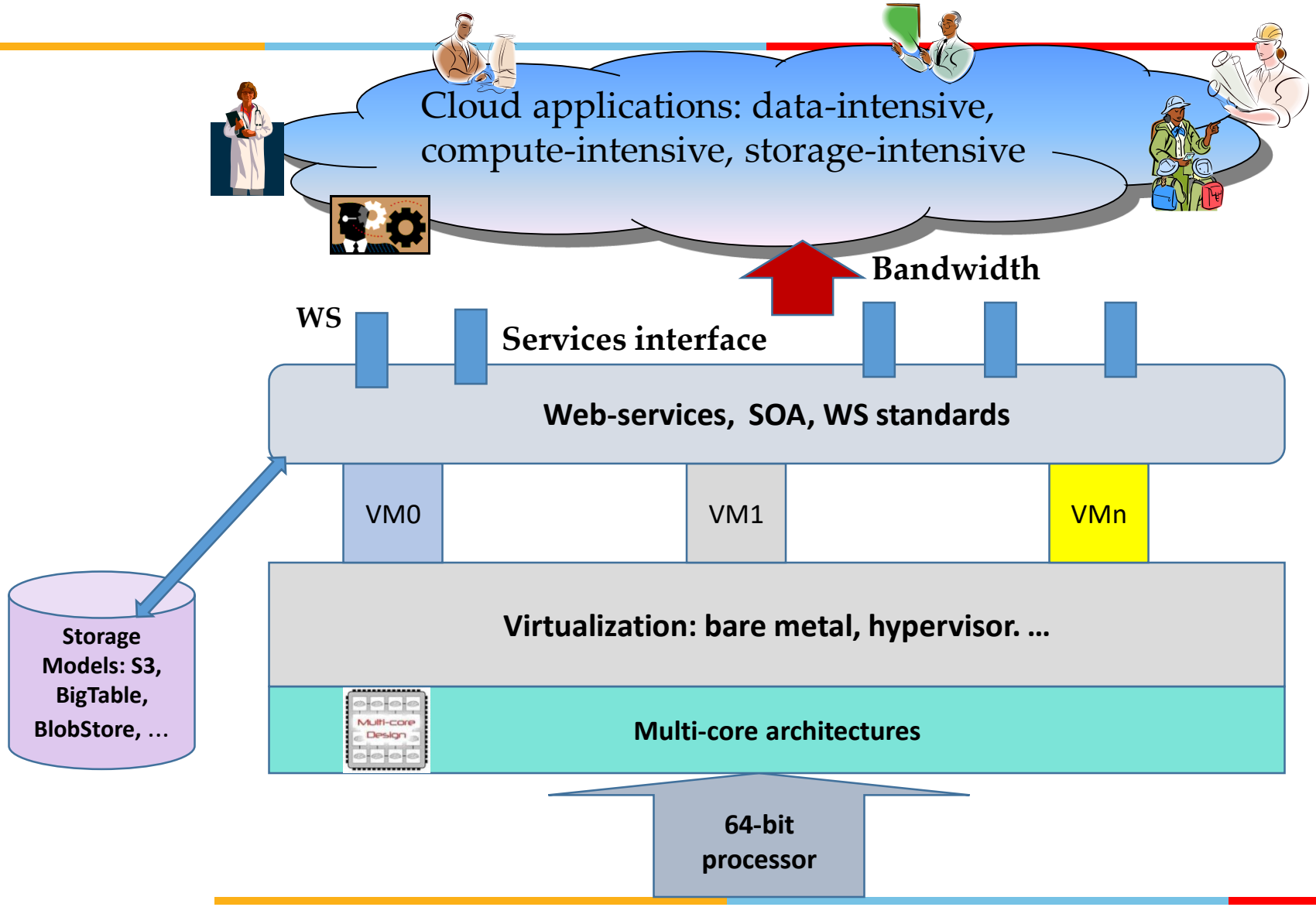
SEWP ZG527

Motivation



1. Web Scale Problems
2. Web 2.0 and Social Networking
3. Information Explosion
4. Mobile Web

Technology Advances



What is Cloud Computing?

Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet,

- basically a step on from Utility Computing
- a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
- Using the Internet for communication and transport provides hardware, software and networking services to clients

These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing cont.…

In addition, the platform provides on demand services, that are always on, anywhere, anytime and any place.

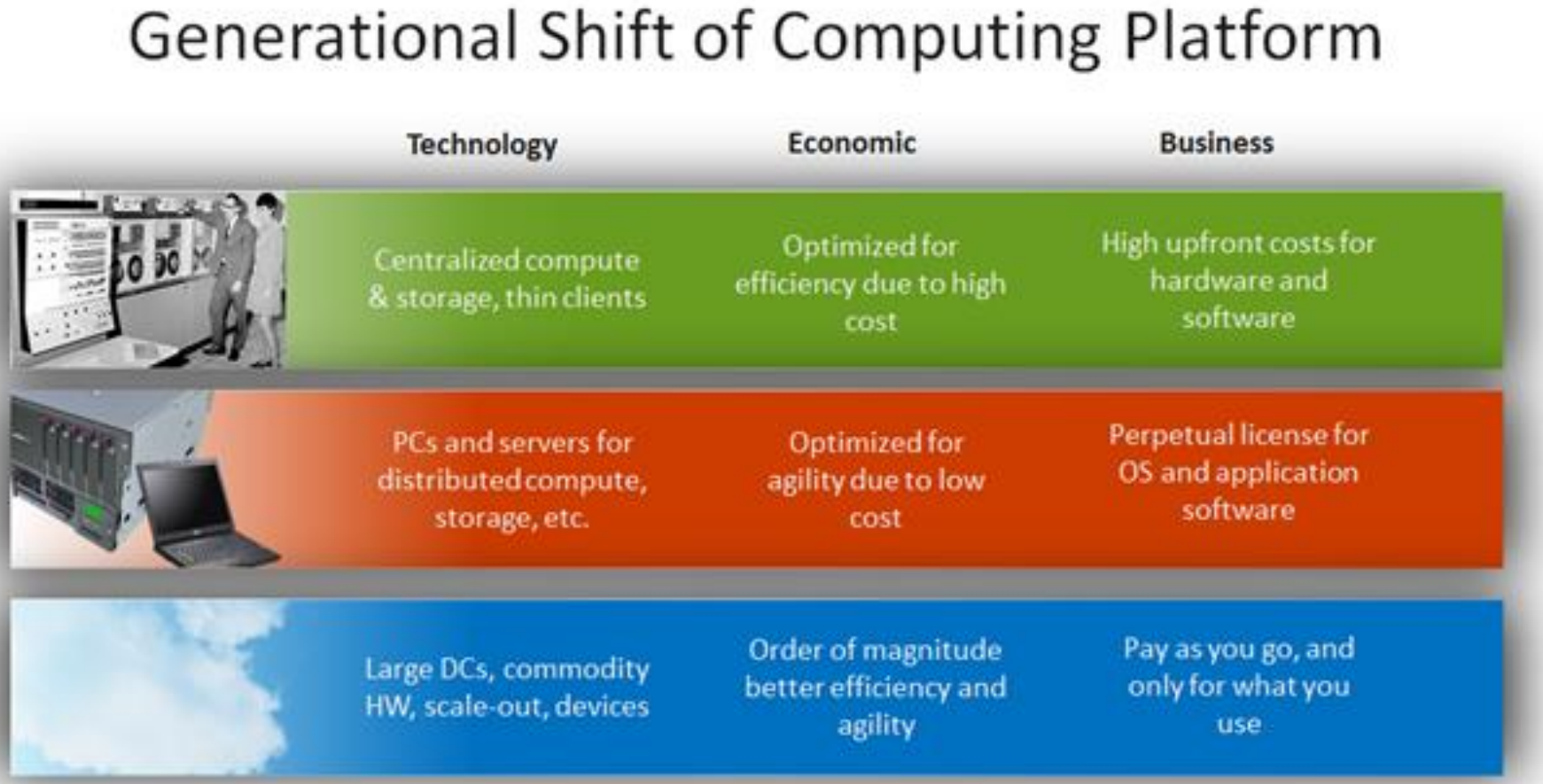
Pay for use and as needed, elastic

- scale up and down in capacity and functionalities

The hardware and software services are available to

- general public, enterprises, corporations and businesses markets

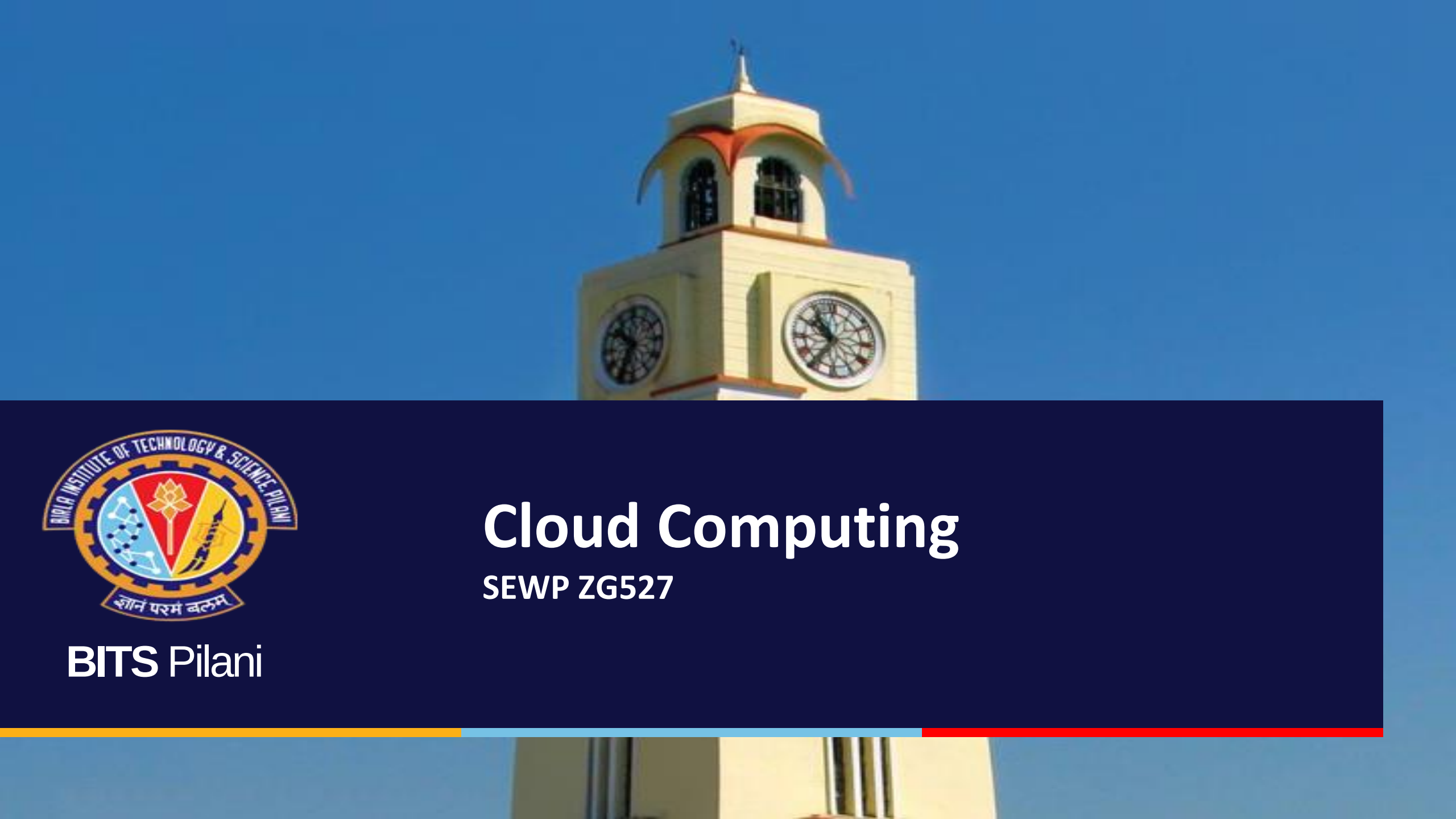
Drivers for the new Platform



http://blogs.technet.com/b/yungchou/archive/2011/03/03/chou-s-theories-of-cloud-computing-the-5-3-2-principle.aspx

Cloud Summary

- Shared pool of configurable computing resources
- On-demand network access
- Provisioned by the Service Provider



BITS Pilani

Cloud Computing

SEWP ZG527

Cloud Computing: Definition

The US National Institute of Standards (NIST) defines cloud computing as follows:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

3-4-5 rule of Cloud Computing

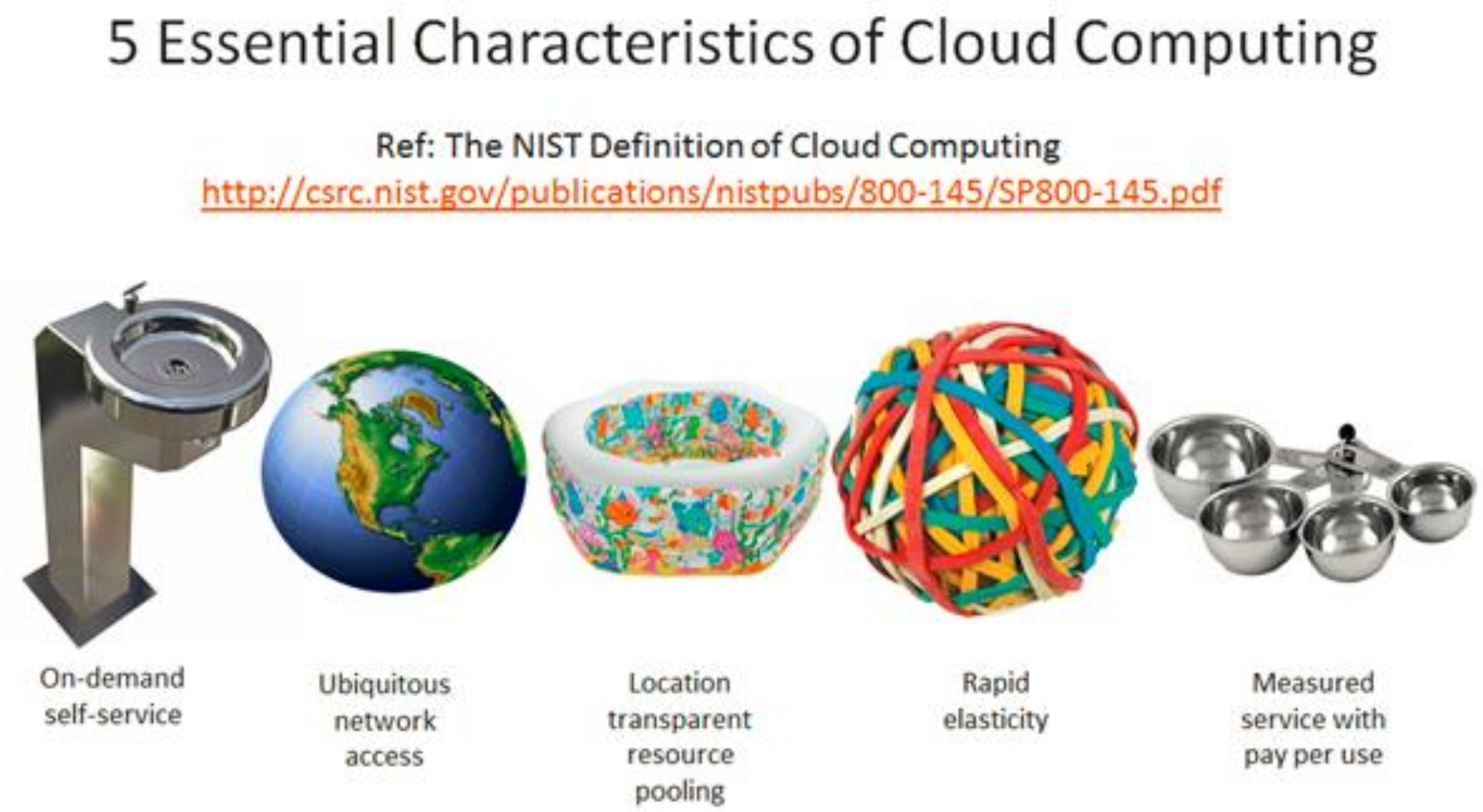
NIST specifies 3-4-5 rule of Cloud Computing

3 cloud service models or service types for any cloud platform

4 deployment models

5 essential characteristics of cloud computing infrastructure

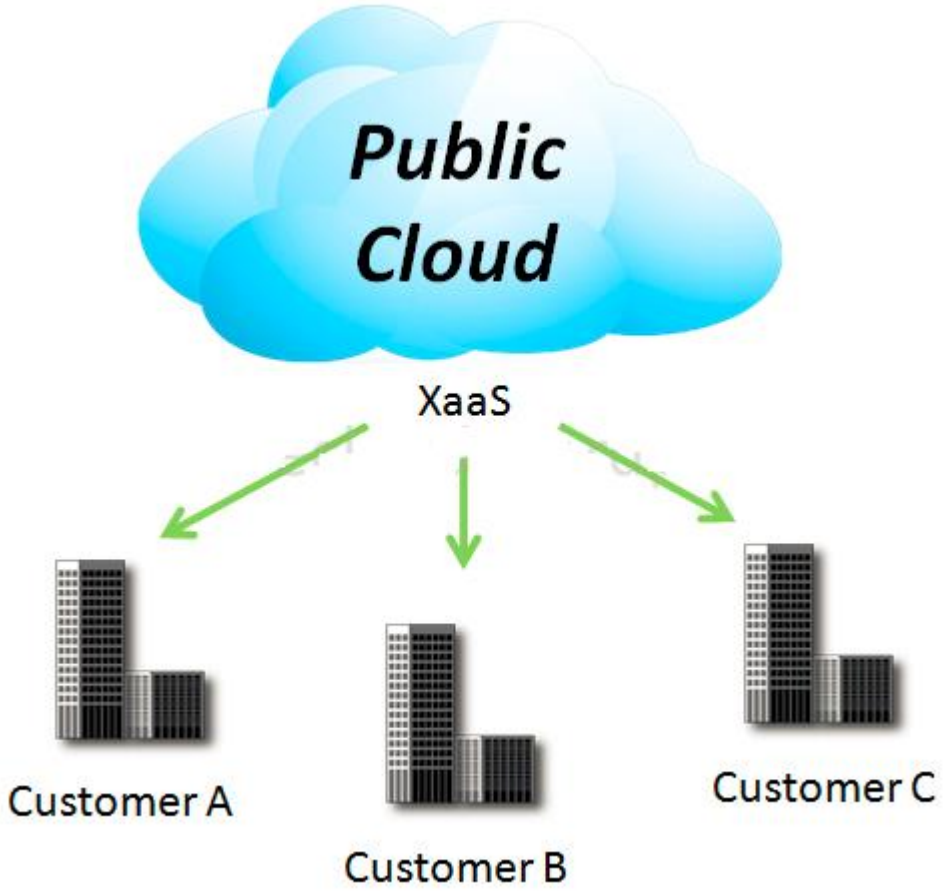
Characteristics of Cloud Computing



- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

4 Deployment Models

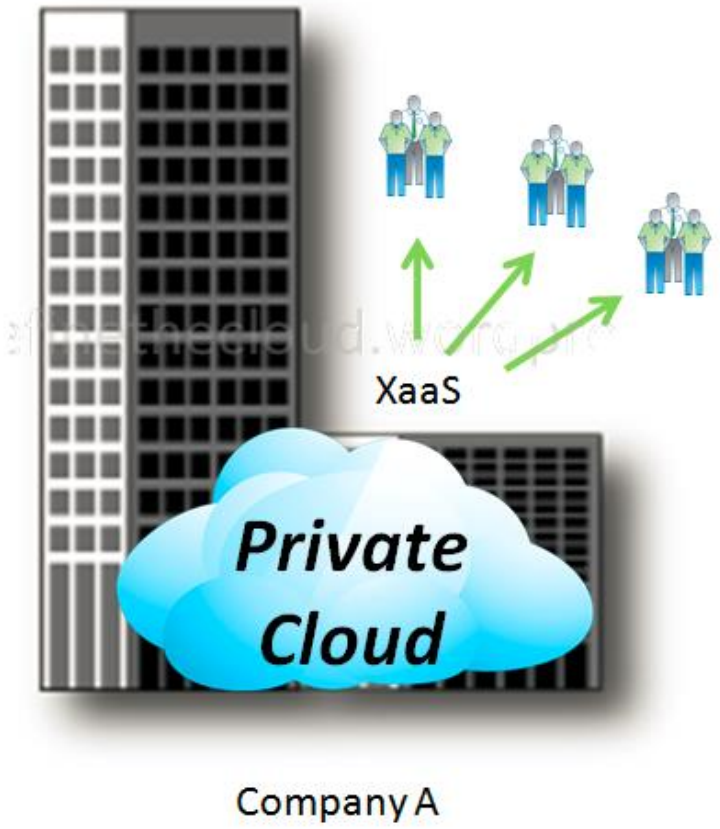
1. Public Cloud



Mega-scale cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

4 Deployment Models

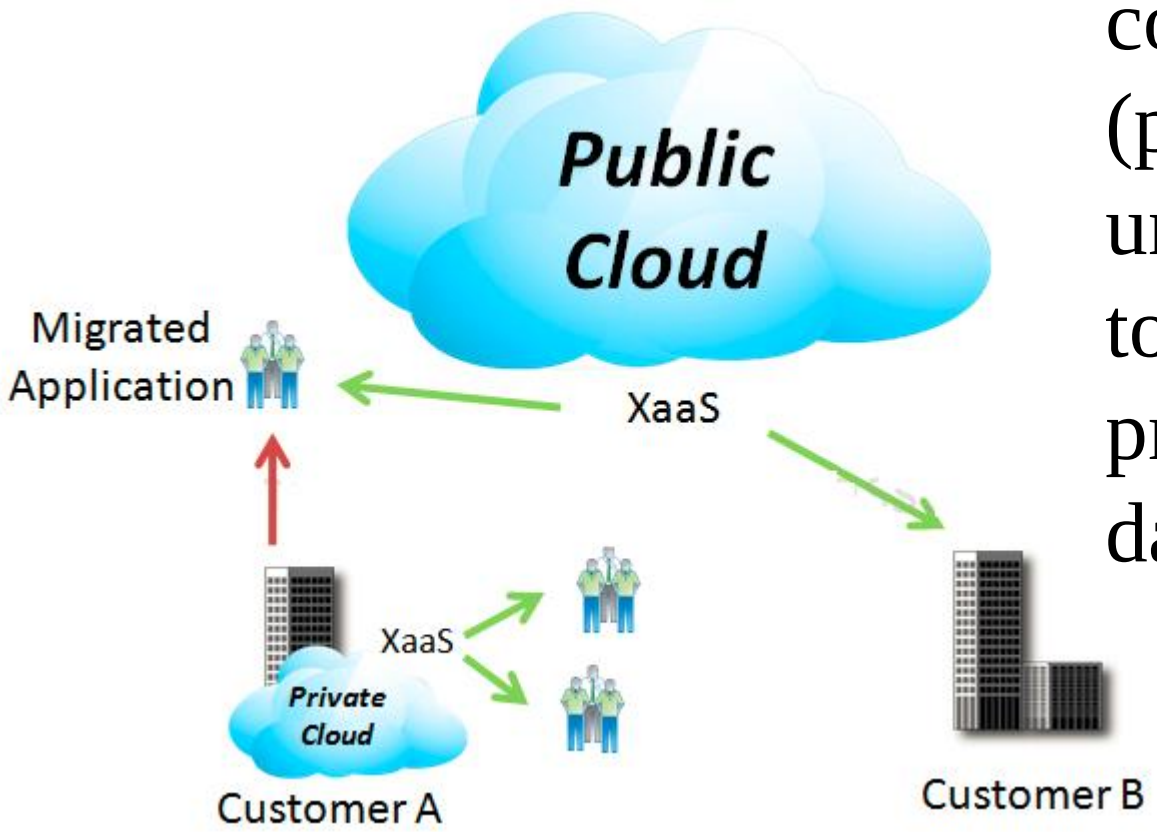
2. Private Cloud



The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

4 Deployment Models

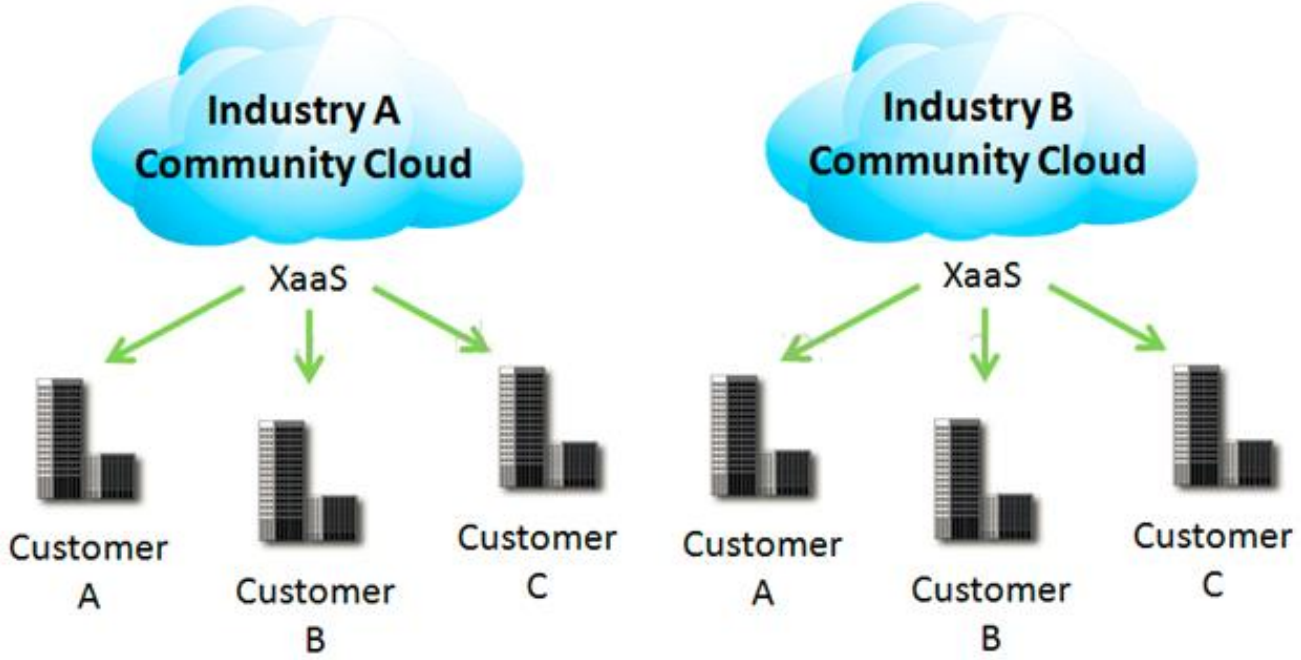
3. Hybrid Cloud



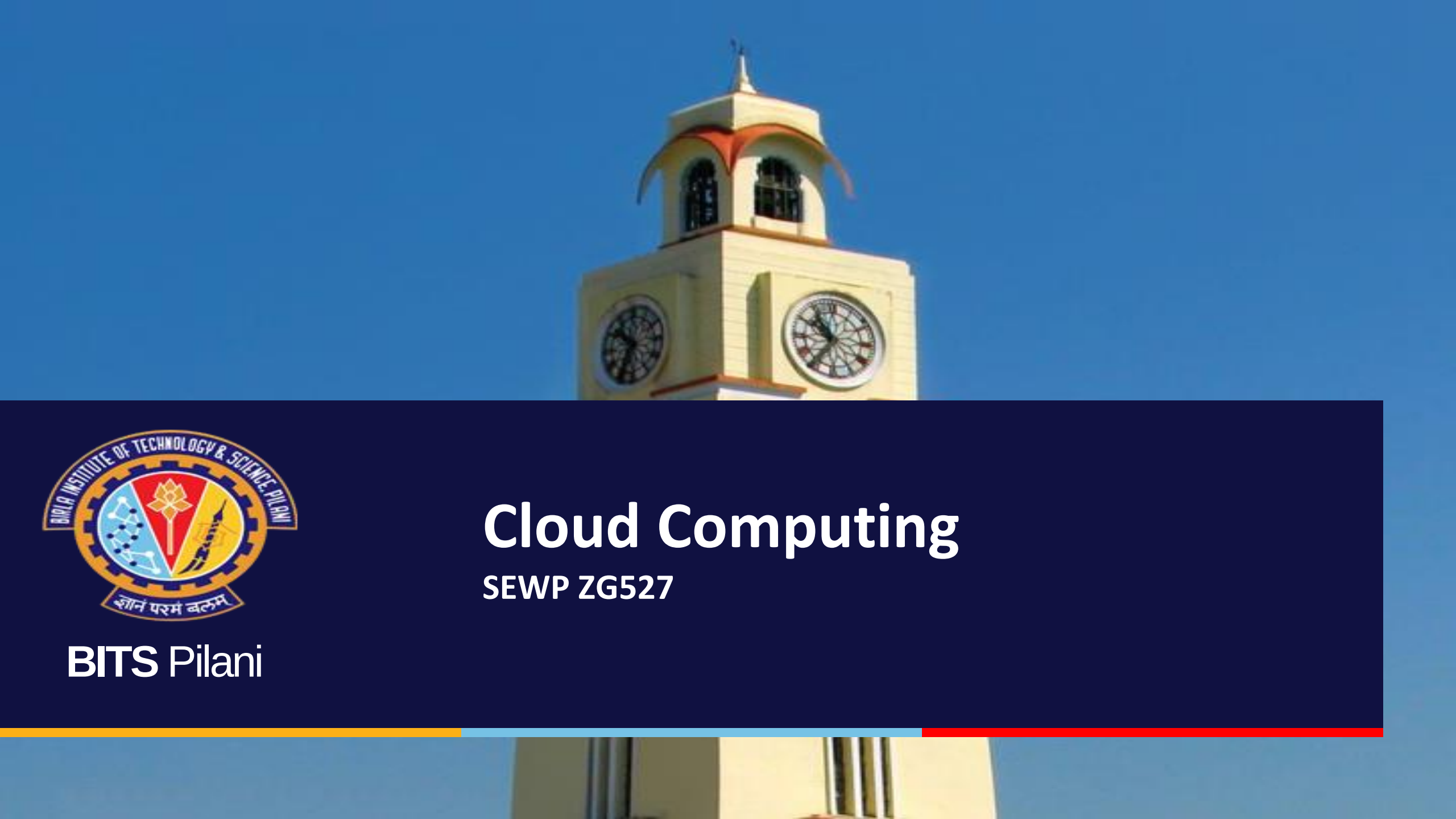
The cloud infrastructure is a composition of two or more clouds (private or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability

4 Deployment Models

4. Community Cloud



Community Clouds are when an ‘infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premise or off premise’ according to NIST.
A community cloud is a cloud service shared between multiple organizations with a common tie/goal/objective.
E.g. OpenCirrus

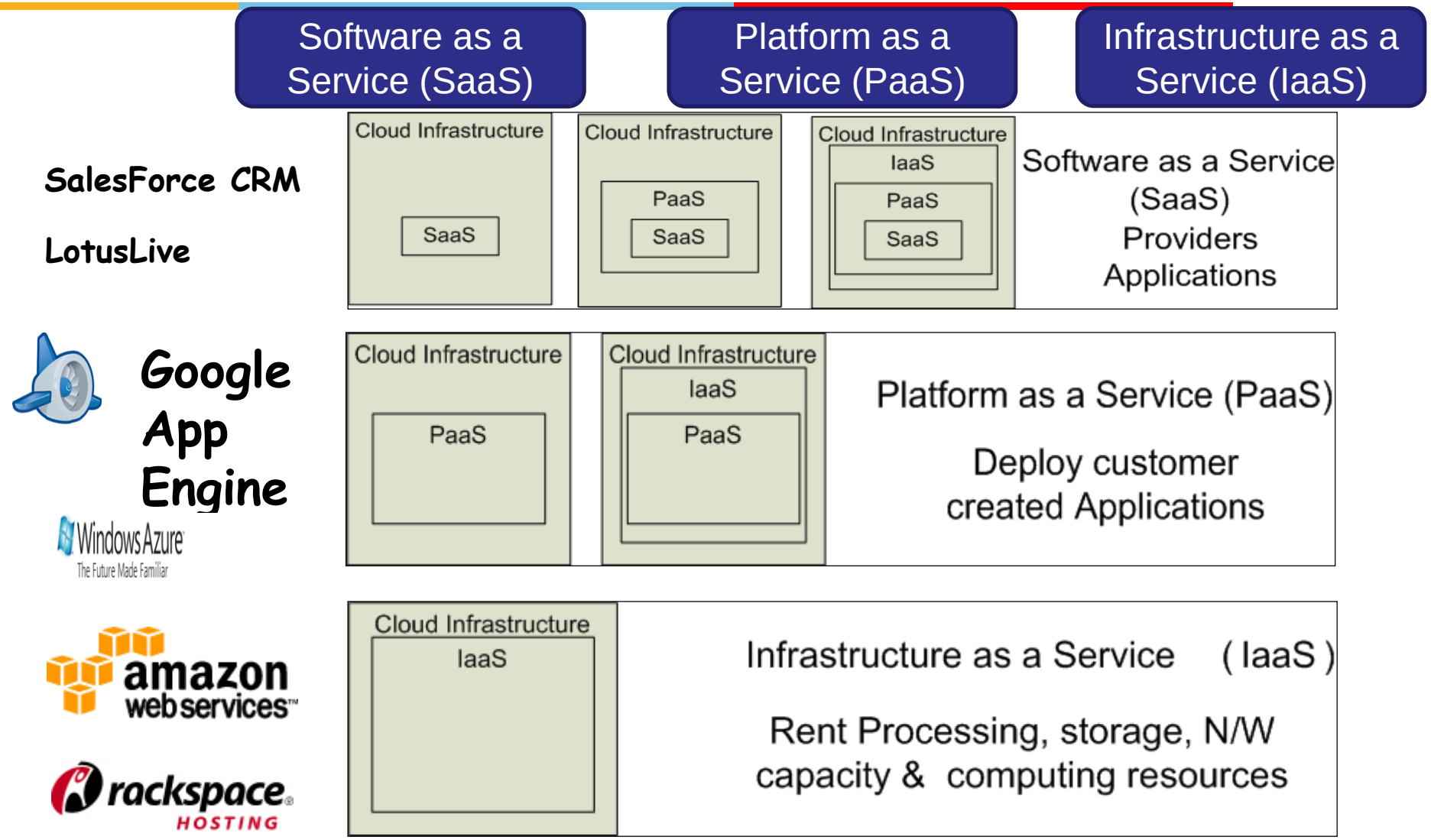


BITS Pilani

Cloud Computing

SEWP ZG527

3 Cloud Service Models



Software as a Service (SaaS)

Software as a service features a complete application
offered as a service on demand.
A single instance of the software runs on the cloud and services multiple end users or client organizations.
E.g. salesforce.com , Google Apps

Platform as a Service

Platform as a service encapsulates a layer of software and provides it as a service that can be used to build higher-level services.

2 Perspectives for PaaS :-

1. Producer:- Someone producing PaaS might produce a platform by integrating an OS, middleware, application software, and even a development environment that is then provided to a customer as a service.

2. Consumer:-Someone using PaaS would see an encapsulated service that is presented to them through an API. The customer interacts with the platform through the API, and the platform does what is necessary to manage and scale itself to provide a given level of service.

Virtual appliances can be classified as instances of PaaS.

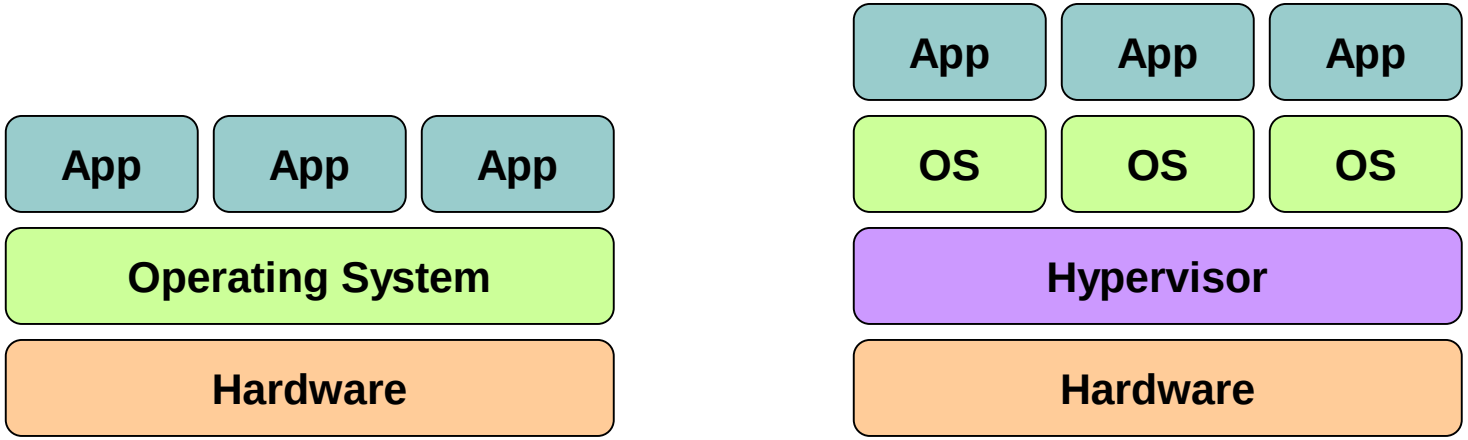
Infrastructure as a Service

Infrastructure as a service delivers basic storage and computing capabilities as standardized services over the network.

Servers, storage systems, switches, routers , and other systems are pooled and made available to handle workloads that range from application components to high-performance computing applications.

Cloud Infrastructures

Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

Cloud Providers Characteristics

- **Provide on-demand provisioning of computational resources**
- **Use virtualization technologies to lease these resources**
- **Provide public and simple remote interfaces to manage those resources**
- **Use a pay-as-you-go cost model, typically charging by the hour**
- **Operate data centers large enough to provide a seemingly unlimited amount of resources to their clients**

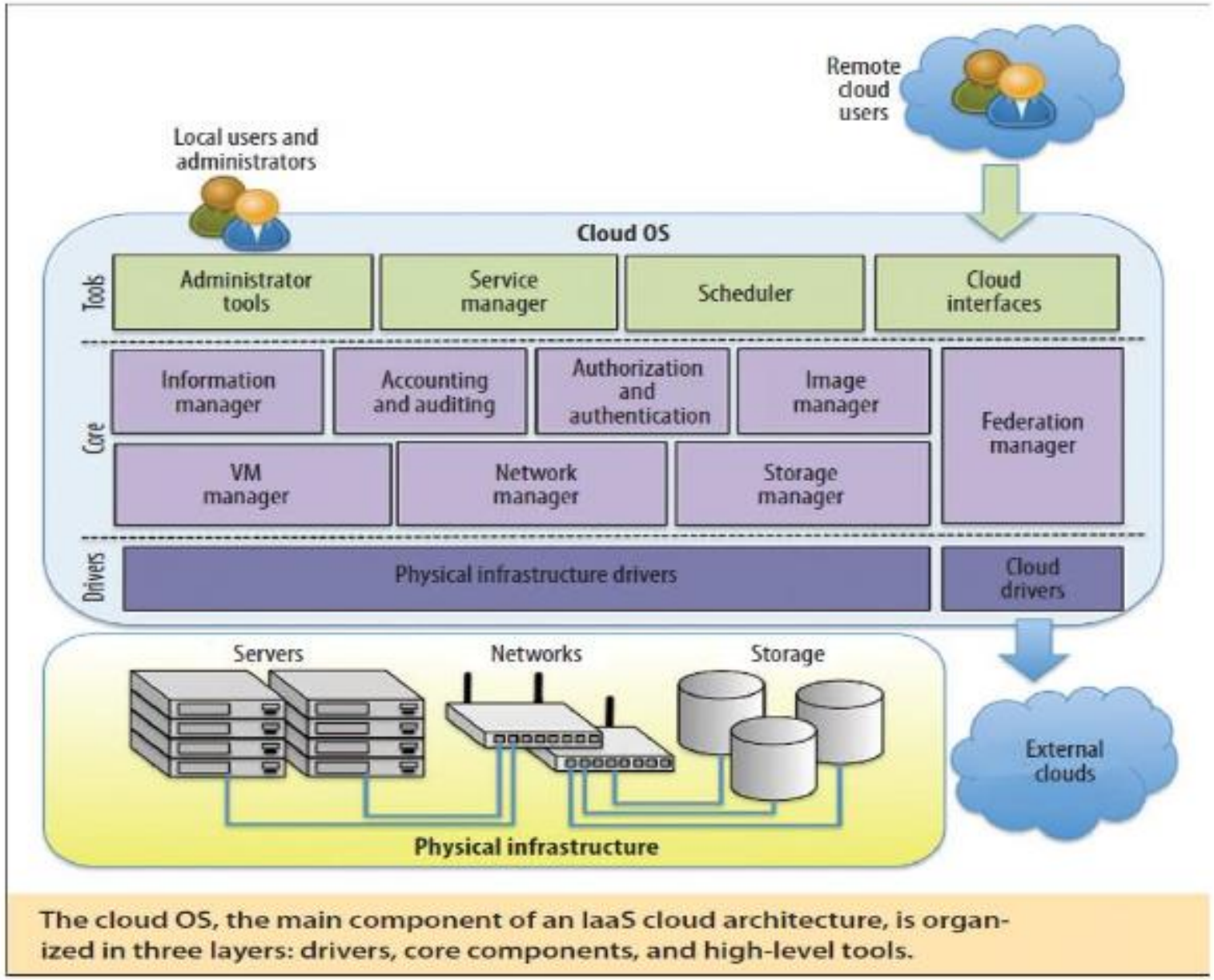
Management of Virtualized Resources

Distributed Management of Virtual Machines

Reservation-Based Provisioning of Virtualized Resources

Provisioning to Meet SLA Commitments

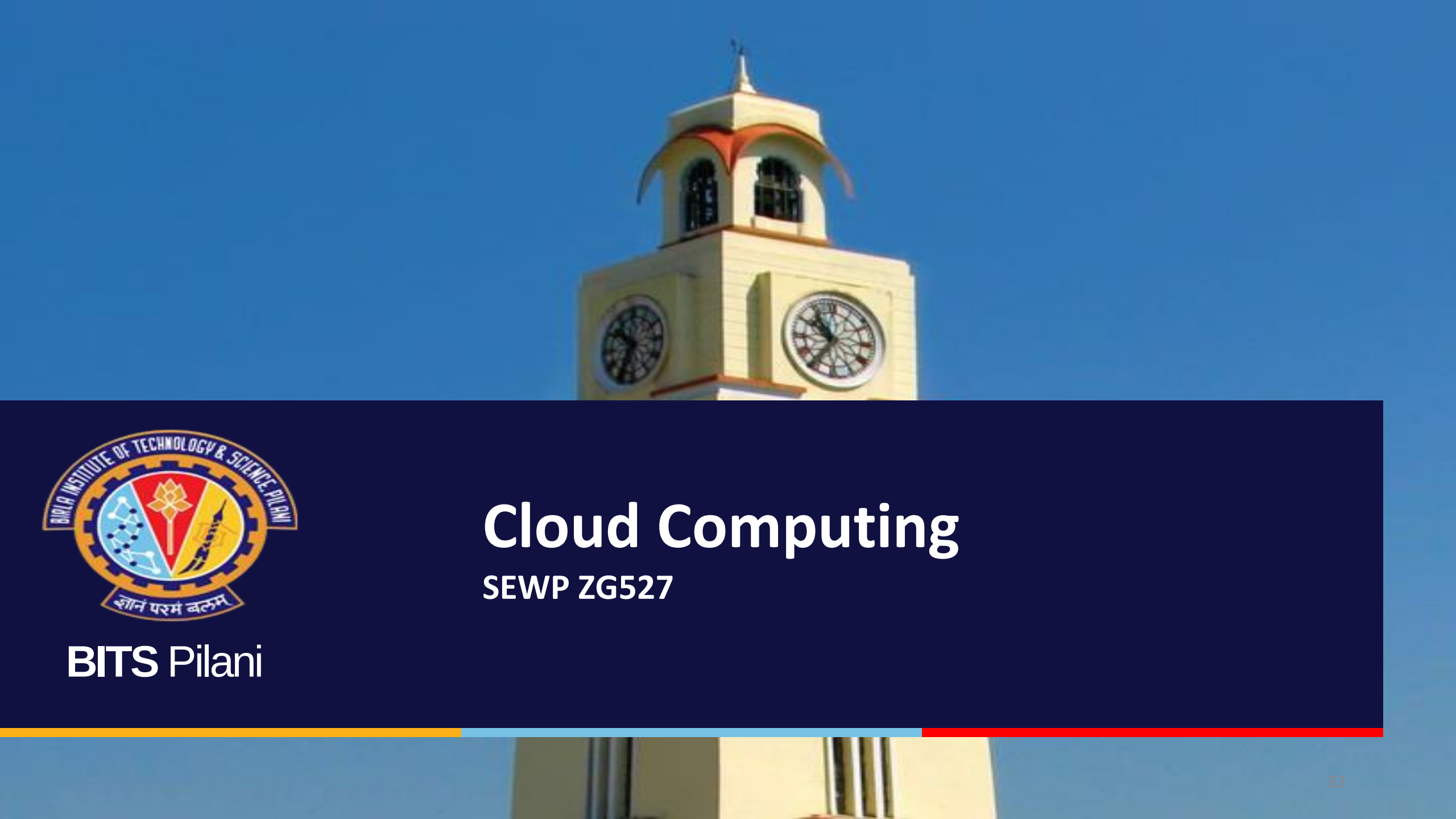
The Cloud OS



The cloud OS, the main component of an IaaS cloud architecture, is organized in three layers: drivers, core components, and high-level tools.

The cloud operating system is responsible for:

1. managing the physical and virtual infrastructure,
2. orchestrating and commanding service provisioning and deployment
3. providing federation capabilities for accessing and deploying virtual resources in remote cloud infrastructures



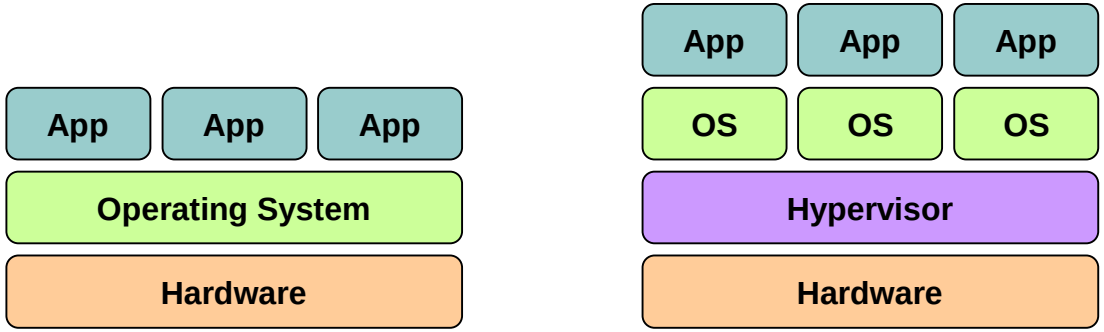
BITS Pilani

Cloud Computing

SEWP ZG527

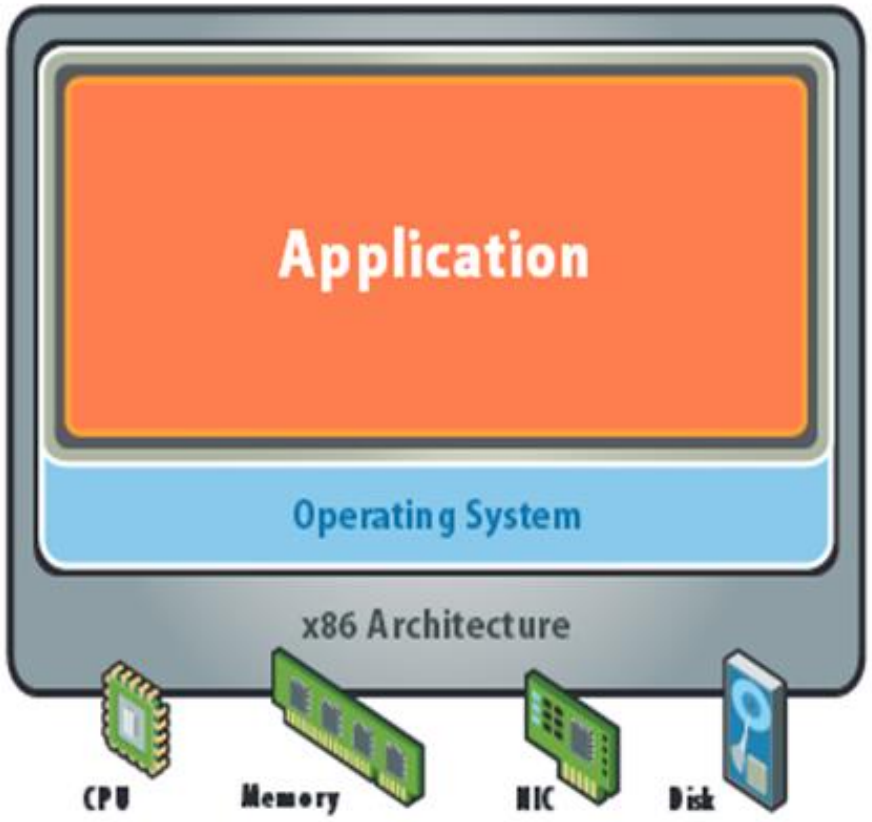
Technology made cloud possible

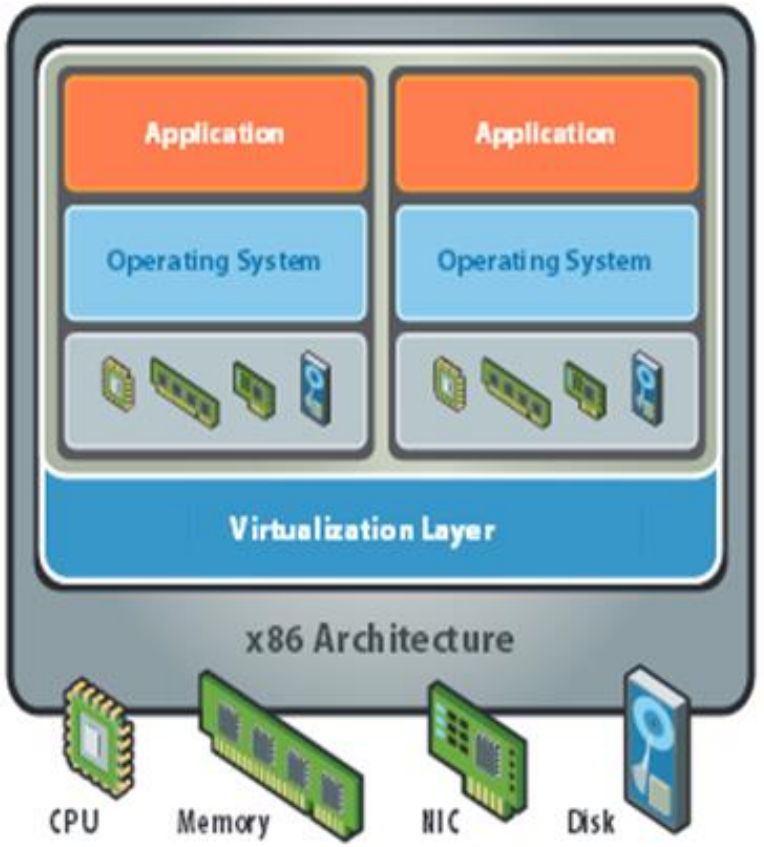
Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

What is Virtualization





What does Virtualization do?

- Virtualization allows multiple operating system instances to run concurrently on a single computer
- It is a means of separating hardware from a single operating system.
- Each “guest” OS is managed by a Virtual Machine Monitor (VMM), also known as a hypervisor.
- Because the virtualization system sits between the guest and the hardware, it can control the guests’ use of CPU, memory, and storage, even allowing a guest OS to migrate from one machine to another.
- Instead of purchasing and maintaining an entire computer for one application, each application can be given its own operating system, and all those operating systems can reside on a single piece of hardware.
- Virtualization allows an operator to control a guest operating system’s use of CPU, memory, storage, and other resources, so each guest receives only the resources that it needs.

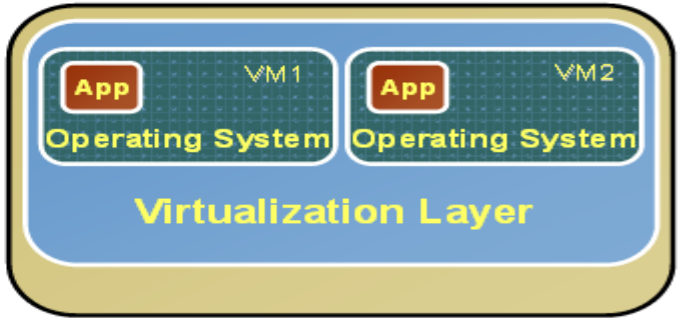
Changes after Virtualization

Before Virtualization

- Single OS image per machine
- Software and hardware tightly coupled
- Running multiple applications on same machine often creates conflict
- Underutilized resources
- Inflexible and costly infrastructure

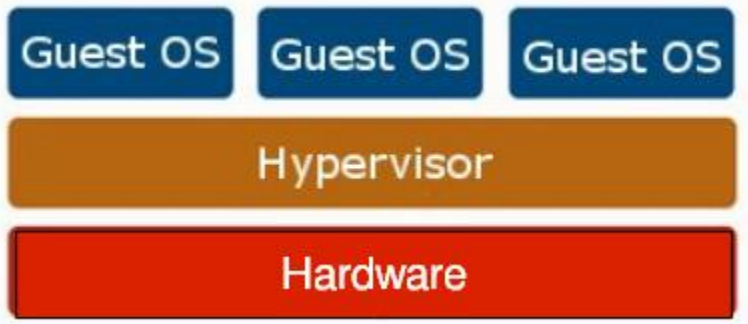
After Virtualization

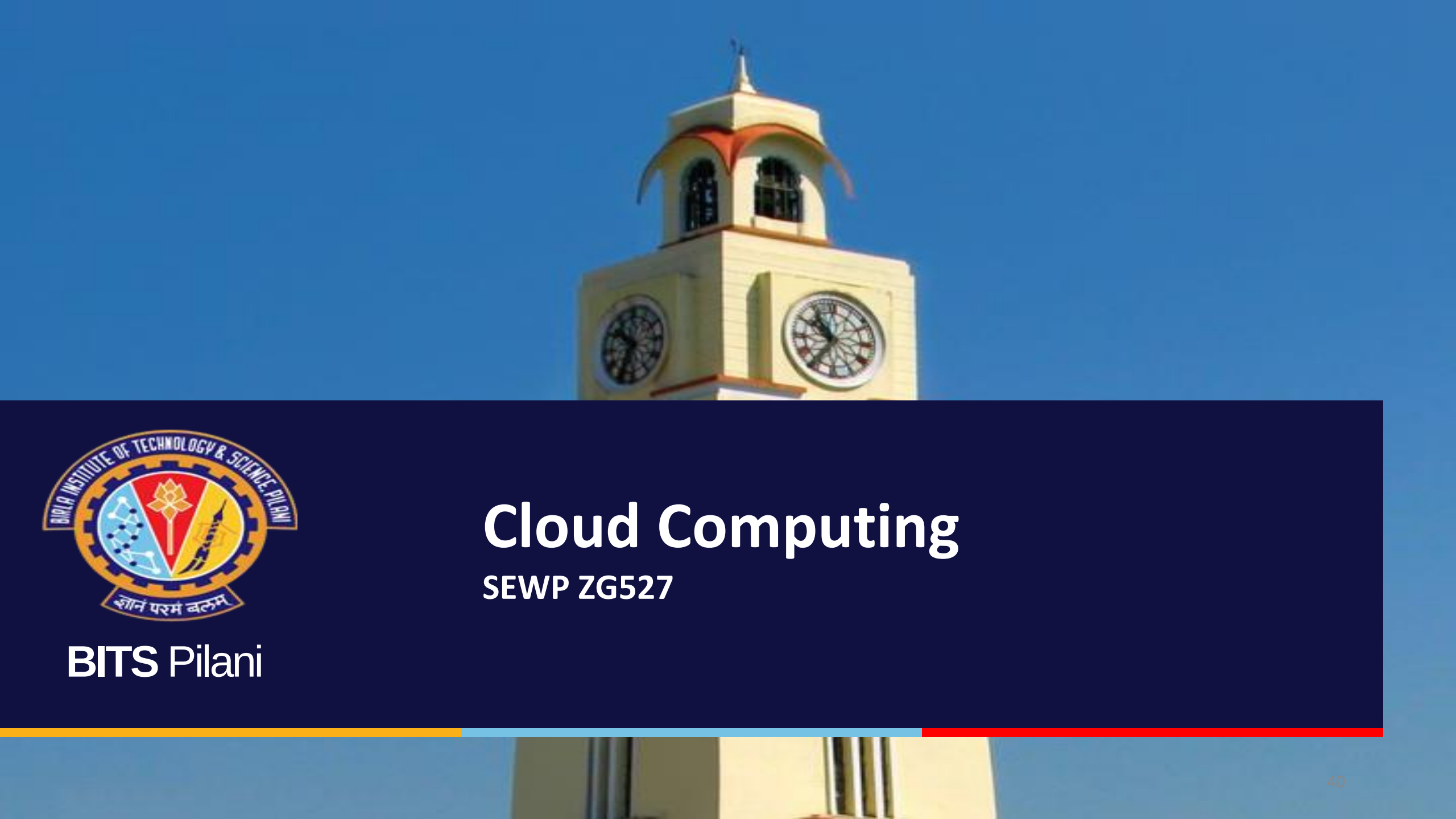
- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines



Virtualization Architecture

- OS assumes complete control of the underlying hardware.
- Virtualization architecture provides this illusion through a hypervisor/VMM.
- Hypervisor/VMM is a software layer which:
  - Allows multiple Guest OS (Virtual Machines) to run simultaneously on a single physical host
  - Provides hardware abstraction to the running Guest OSs and efficiently multiplexes underlying hardware resources





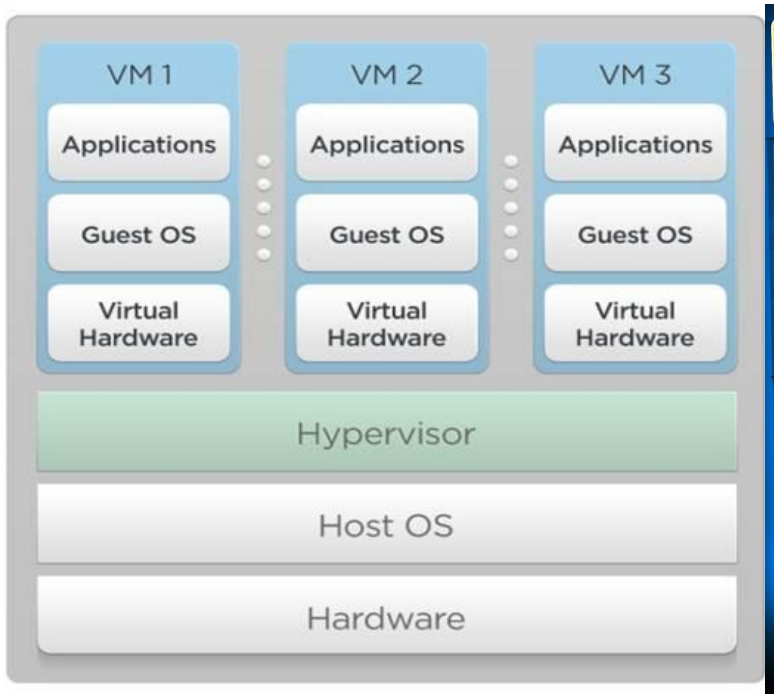
BITS Pilani

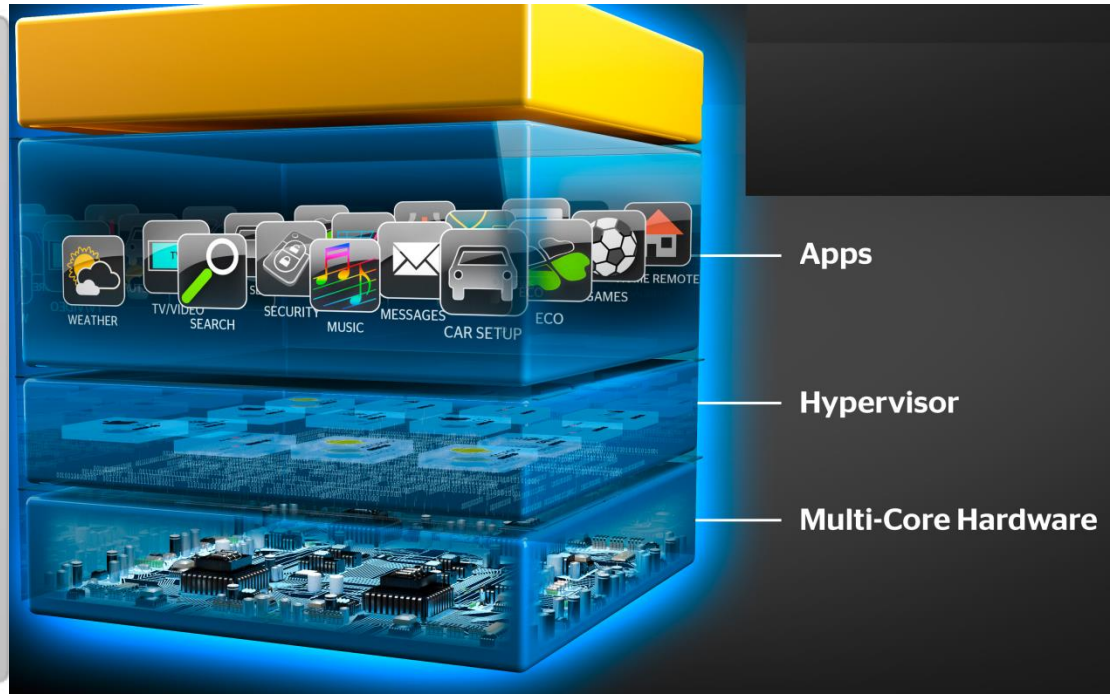
Cloud Computing

SEWP ZG527

Hypervisor

A thin layer of software that generally provides virtual partitioning capabilities which runs directly on hardware, but underneath higher-level virtualization services. Sometimes referred to as a “bare metal” approach.





Hypervisor Design Goals

- Isolation
  - Security isolation
  - Fault isolation
  - Resource isolation
- Reliability
  - Minimal code base
  - Strictly layered design
  - Not extensible
- Scalability
  - Scale to large number of cores
  - Large memory systems

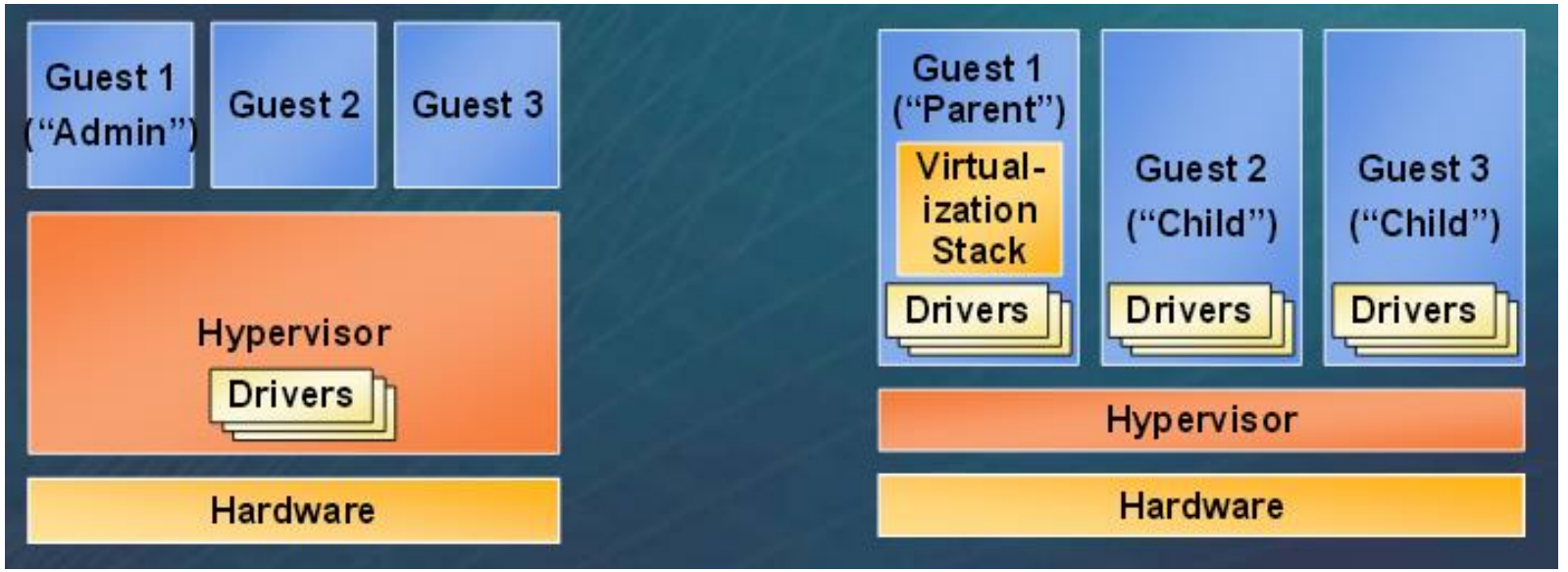
How Hypervisor goals are achieved?

- Partitioning Kernel
  - “Partition” is isolation boundary
  - Few virtualization functions; relies on virtualization stack
- Very thin layer of software
  - Microkernel
  - Highly reliable
  - Basis for smaller Trusted Computing Base (TCB)
- No device drivers
  - Drivers run in a partition
- Well-defined interface
  - Allow others to create support for their OSes as guests

Hypervisor

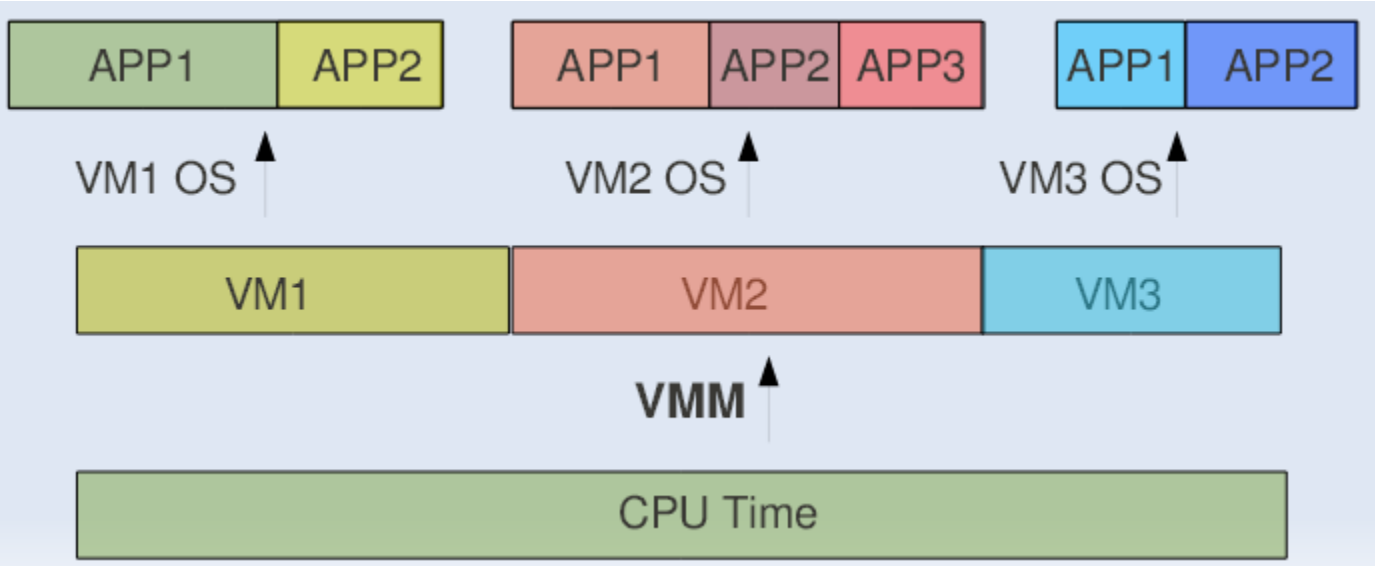
Monolithic versus Microkernelized

- Monolithic hypervisor
  - Simpler than a modern kernel, but still complex
  - Contains its own drivers model
- Microkernelized hypervisor
  - Simple partitioning functionality
  - Increase reliability and minimize lowest level of the TCB
  - No third-party code
  - Drivers run within guests



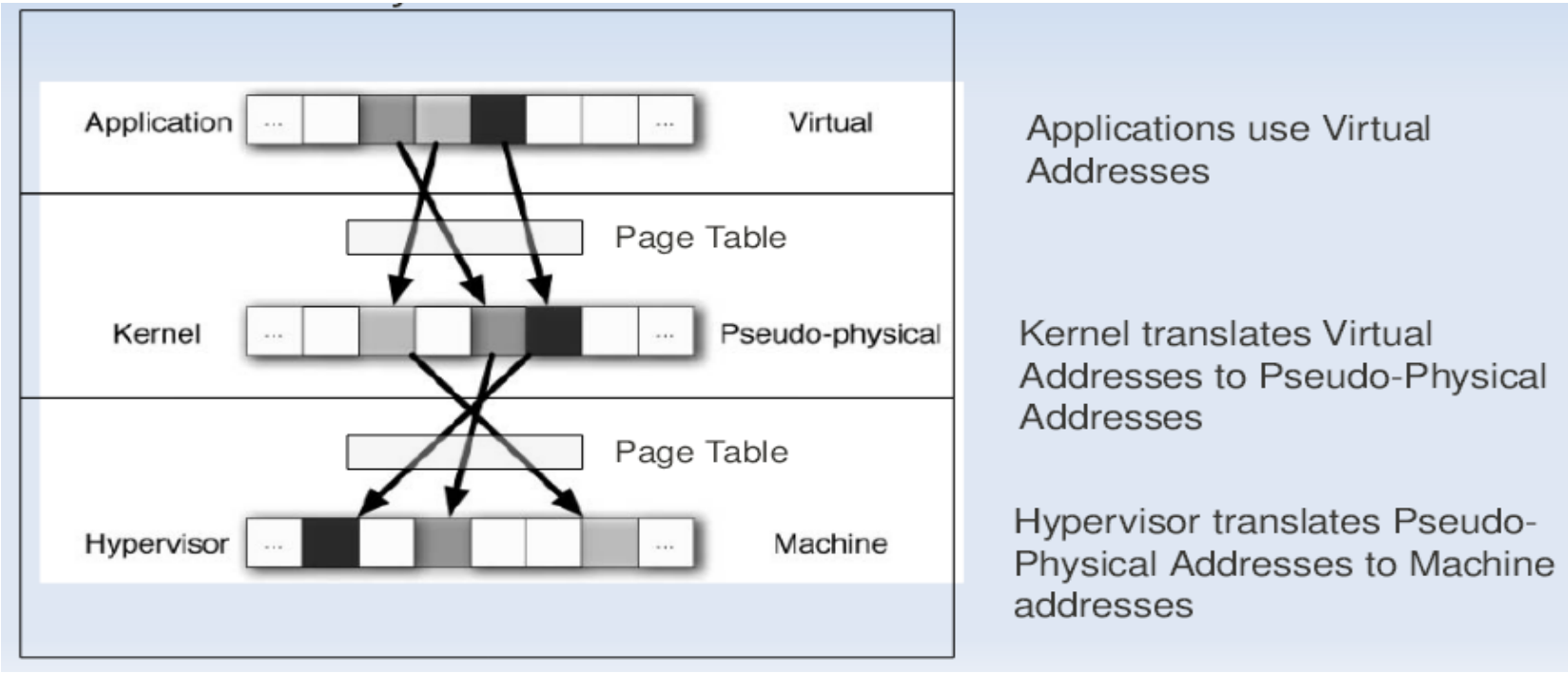
CPU Sharing

- VMM or Hypervisor provides a virtual view of CPU to VMs.
- In multi processing, CPU is alloted to the different processes in form of time slices by the OS.
- Similarly VMM or Hypervisor allots CPU to different VMs.



Memory Sharing

- In Multiprogramming there is a single level of indirection maintained by Kernel.
- In case of Virtual Machines there is one more level of indirection maintained by VMM

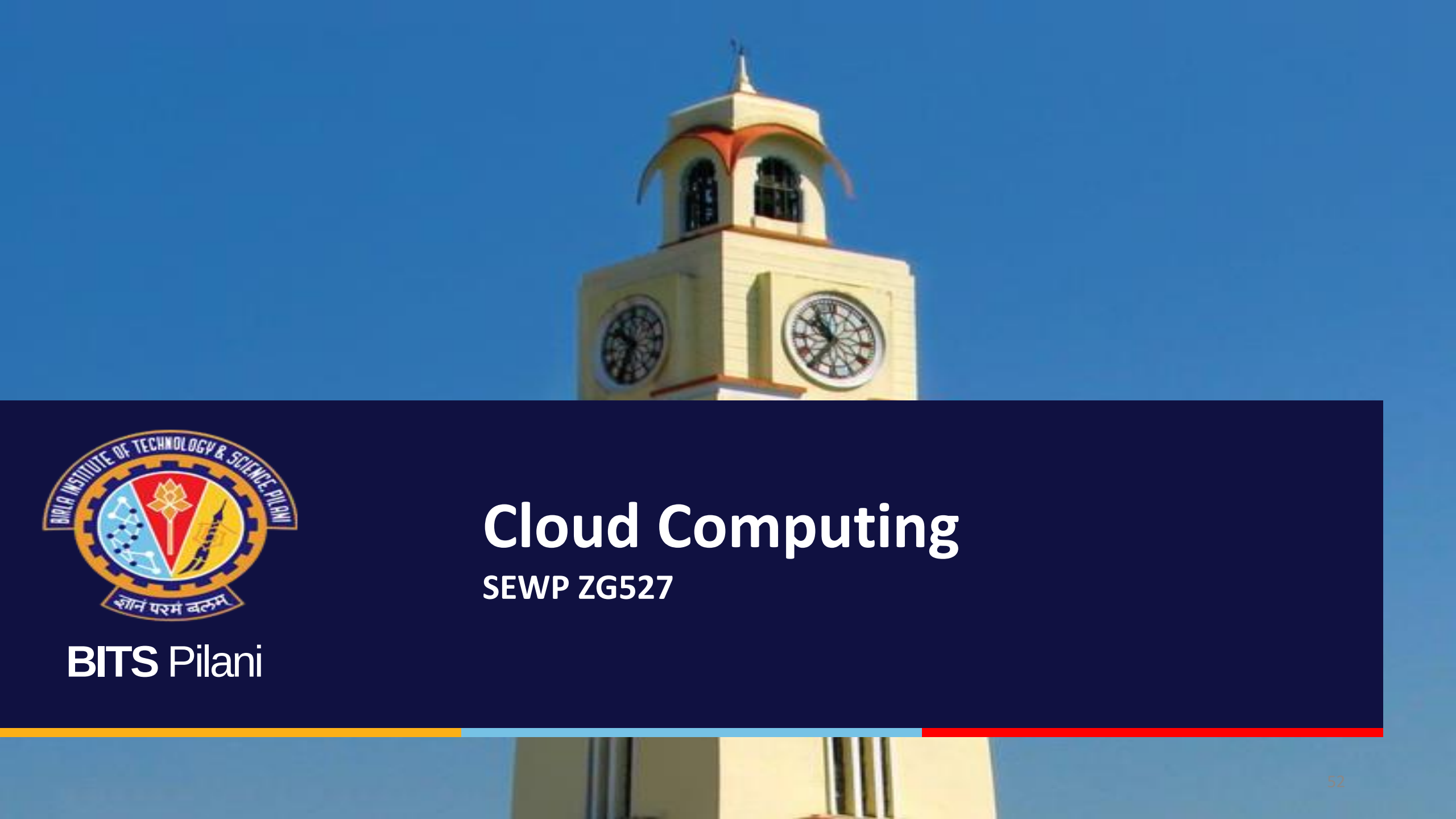


IO Sharing

- Device needs to use Physical Memory location.
- In a virtualized environment, the kernel is running in a hypervisor-provided virtual address space
- Allowing the guest kernel to convey an arbitrary location to device for writing is a serious security hole
- Each device defines its own protocol for talking to drivers

Thanks!!!

Queries?



BITS Pilani

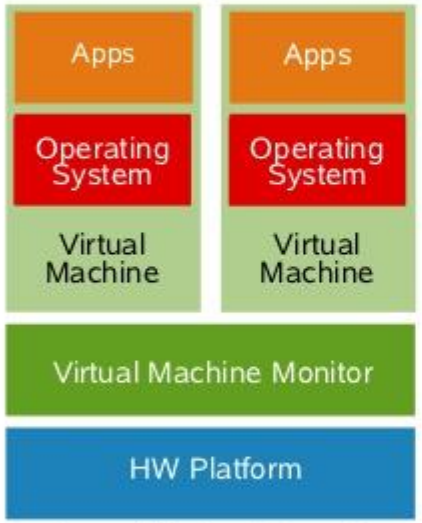
Cloud Computing

SEWP ZG527

Approaches for Virtualization

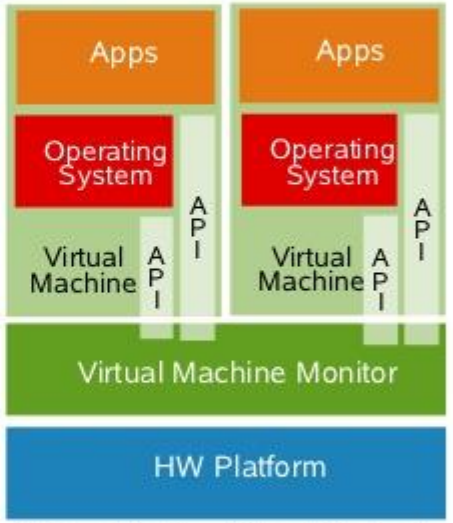
Full & Paravirtualization Overview

Full Virtualization



Runtime modification of Guest OS: VMM manages the conflict, then returns to OS

Paravirtualization



Static modification of Guest OS prior to runtime: Privileged instruction calls are exchanged with API functions provided by the VMM
- Almost no performance degradation
- Significant scalability

13 © Novell Inc. All rights reserved

Full Virtualization

❑ Full virtualization

- In its basic form known as “full virtualization” the hypervisor provides a fully emulated machine in which an operating system can run. VMWare is a good example.
- The biggest advantage to this approach is its flexibility: one could run a RISC-based OS as a guest on an Intel-based host.
- While this is an obvious approach, there are significant performance problems in trying to emulate a complete set of hardware in software.

❑Paravirtualization

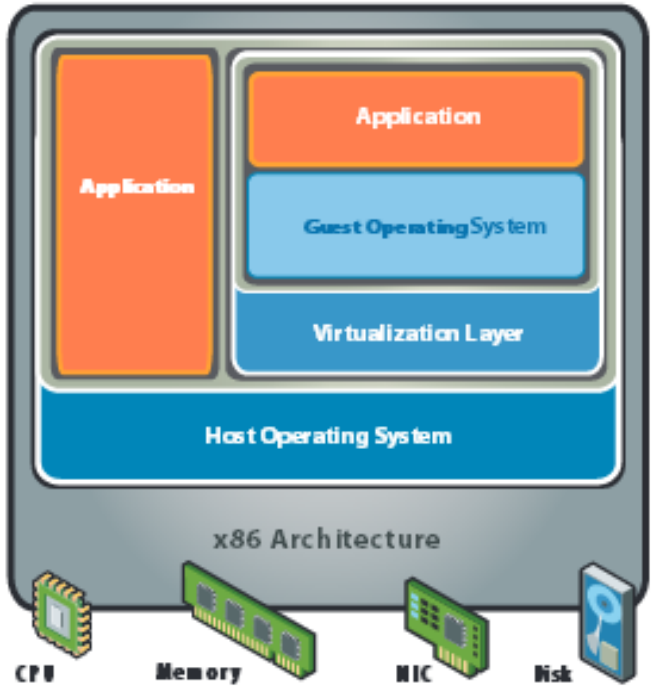
- “Paravirtualization,” found in the XenSource, open source Xen product, attempts to reconcile these two approaches. Instead of emulating hardware, paravirtualization uses slightly altered versions of the operating system which allows access to the hardware resources directly as managed by the hypervisor.
- This is known as hardware-assisted virtualization, and improves performance significantly.
- In order to retain flexibility, the guest OS is not tied to its host OS. Drastically different operating systems can be running in a hypervisor at the same time, just as they can under full virtualization.
- In this way, paravirtualization can be thought of as a low-overhead full virtualization

❑Single Kernel Image (SKI),

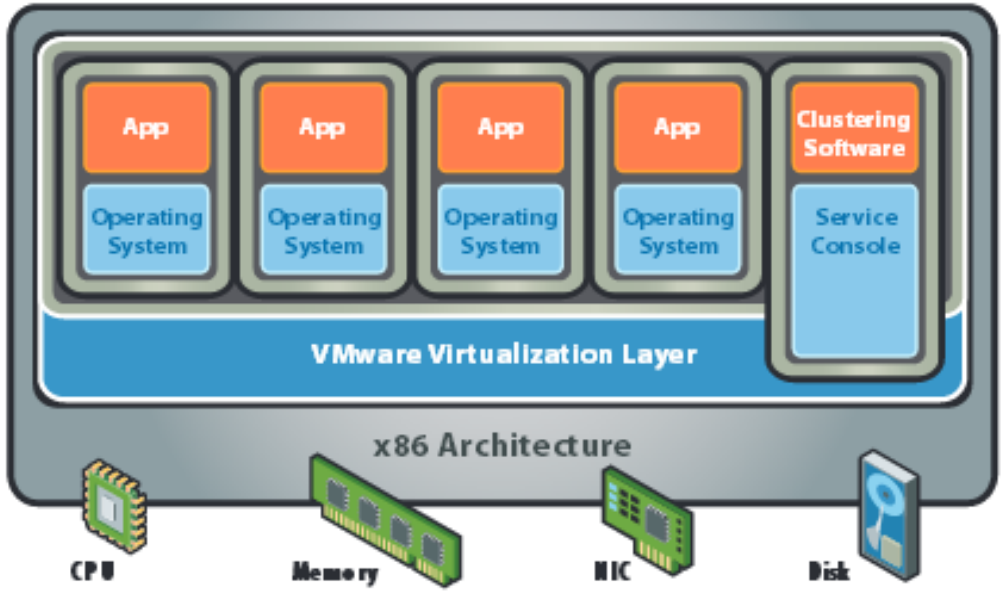
- Single Kernel Image (SKI), in which the host OS spawns additional copies of itself. This kind of virtualization can be found in Swsoft Virtuozzo and Sun Solaris, Zones. SKI can be thought of as “lightweight” virtualization.
- While this approach avoids the performance problems with pure emulation, it does so at the expense of flexibility.
- It is not possible, for instance, to run different versions or even different patch levels of a particular operating system on the same machine.
- Whatever versions exist in the host, that same software will be provided in the guest. SKI also sacrifices the security and reliability provided by other virtualization methods.

- For Industry-standard x86 systems, the two approaches typically used with software-based partitioning are
  - hosted and
  - hypervisor architectures
- A hosted approach provides partitioning services on top of a standard operating system and supports the broadest range of hardware configurations.
- In contrast, a hypervisor architecture is the layer of software installed on a clean x86-based system (hence it is often referred to as a “bare metal” approach). Since it has direct access to the hardware resources, a hypervisor is more efficient than hosted architectures, enabling greater scalability, robustness and performance

x86 Hardware Virtualization



Hosted Architecture



Bare-Metal (Hypervisor) Architecture

Advantages of Virtualization

- Instant provisioning - fast scalability
- Live Migration is possible
- Load balancing and consolidation in a Data Center is possible.
- Low downtime for maintenance
- Virtual hardware supports legacy operating systems efficiently
- Security and fault isolation

Issues to be aware of

- **Software licensing**

  One of the most significant virtualization-related issues to be aware of is software licensing. Virtualization makes it easy to create new servers, but each VM requires its own separate software license. Organizations using expensive licensed applications could end up paying large amounts in license fees if they do not control their server sprawl.

- **IT training**

  IT staff used to dealing with physical systems will need a certain amount of training in virtualization. Such training is essential to enable the staff to debug and troubleshoot issues in the virtual environment, to secure and manage VMs, and to effectively plan for capacity.

- **Hardware investment**

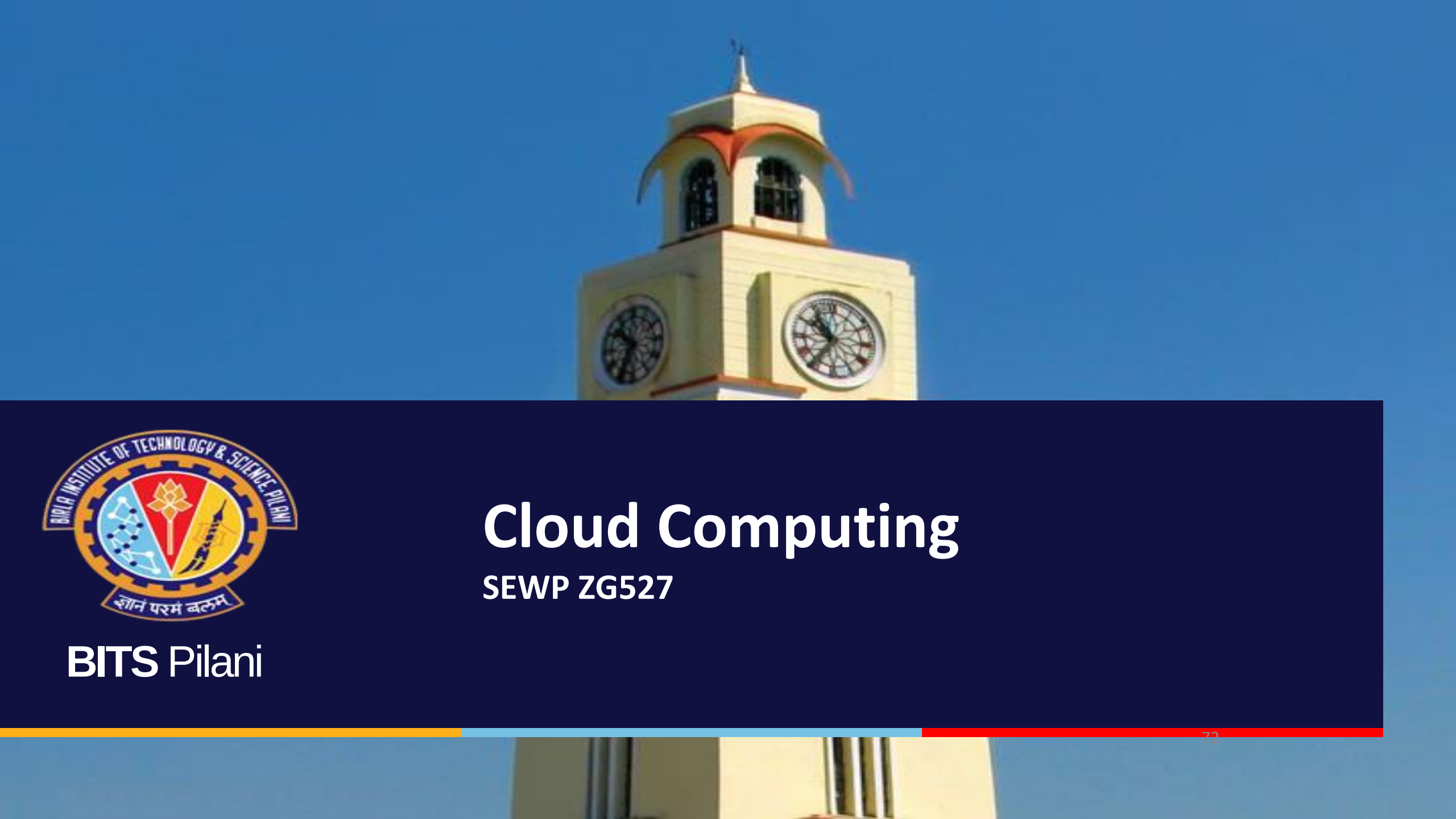
  Server virtualization is most effective when powerful physical machines are used to host several VMs. This means that organizations that have existing not-so-powerful hardware might still need to make upfront investments in acquiring new physical servers to harvest the benefits of virtualization

Issues to be aware of

- Performance can be a concern, especially for in-band deployments, where the virtualization controller or appliance can become a bandwidth bottleneck.
- Interoperability among vendor products is still evolving.
- Failure of the virtualization device, leading to loss of the mapping table.

Thanks!!!

Queries?



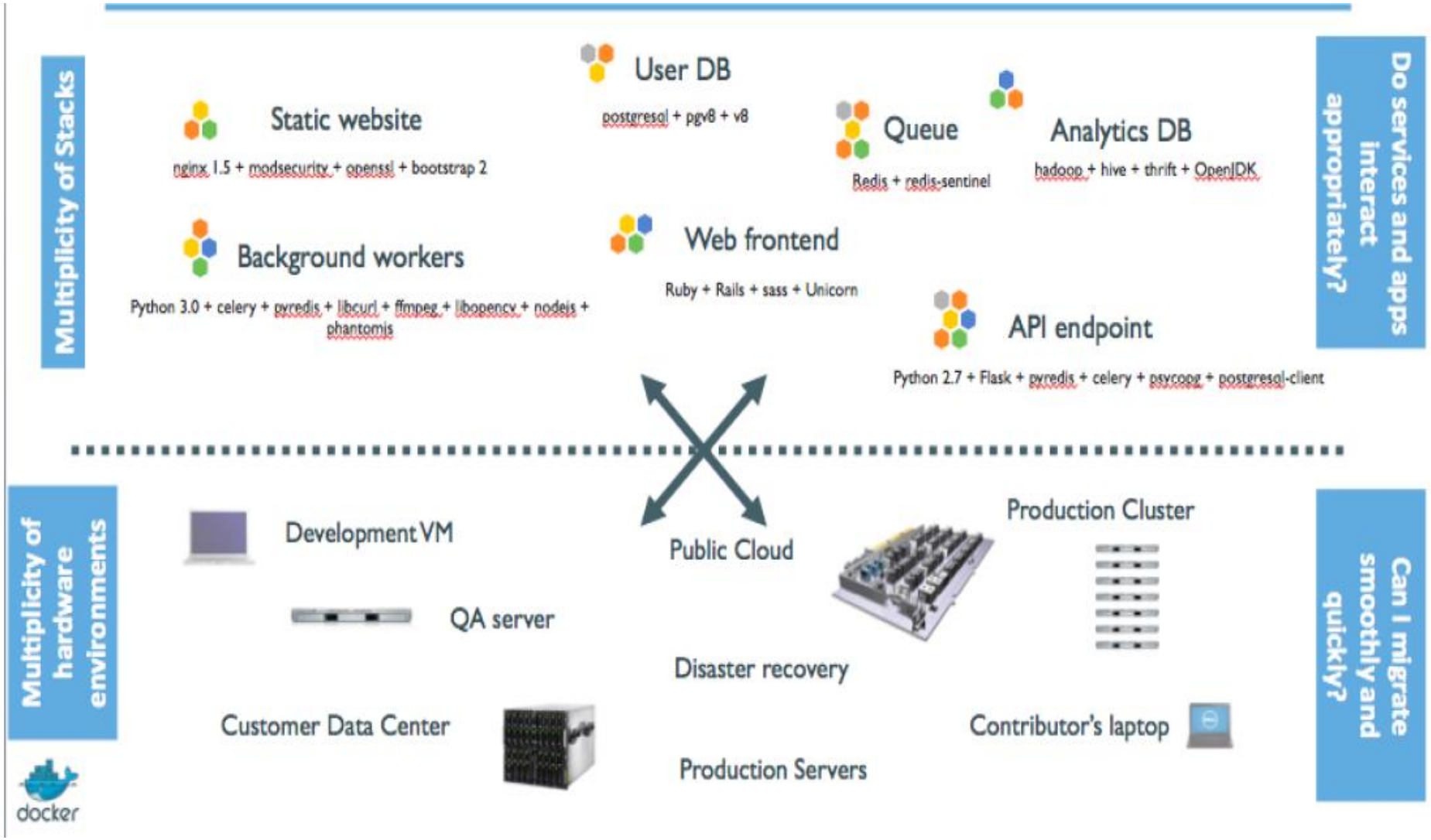
BITS Pilani

Cloud Computing

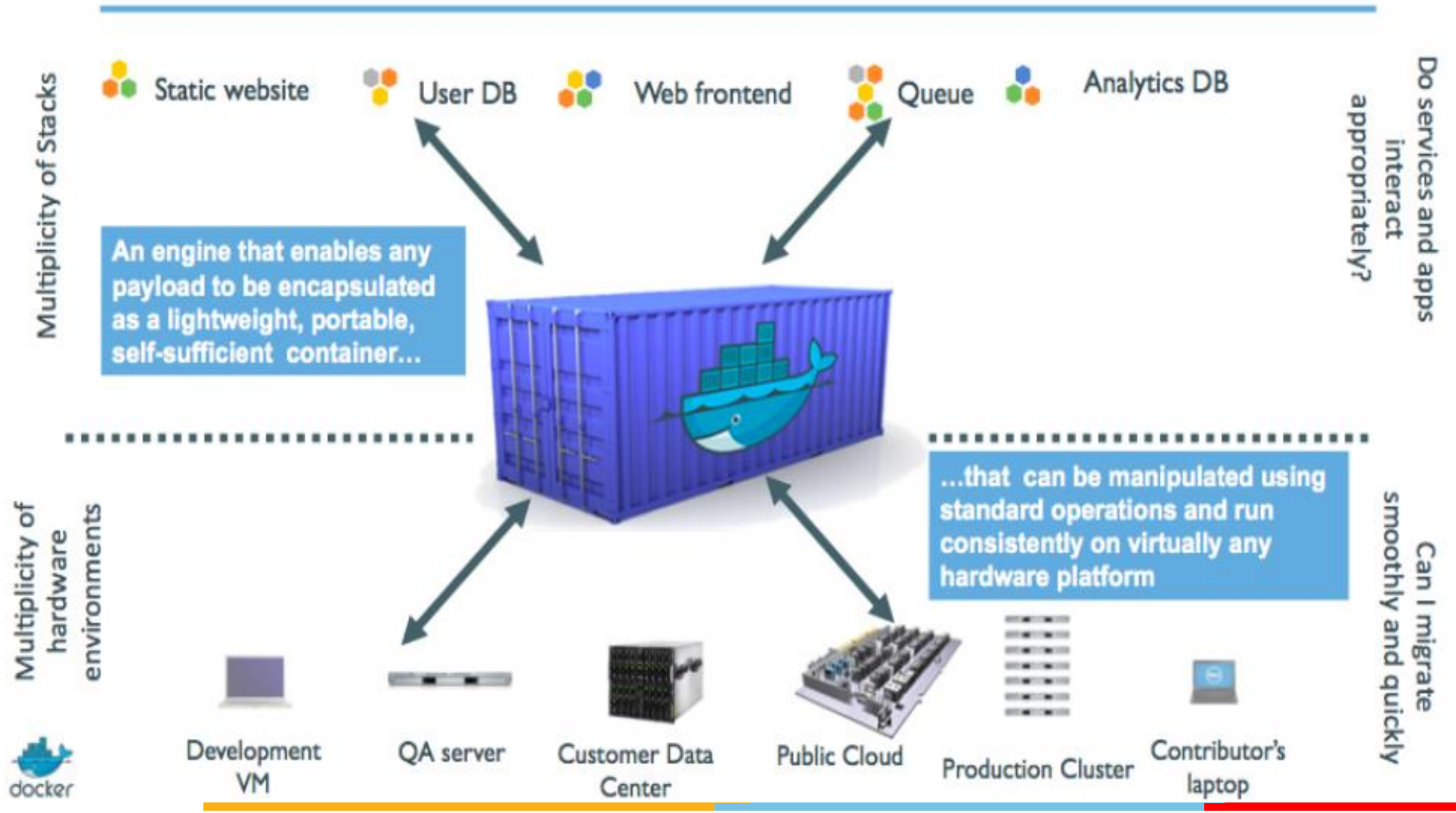
SEWP ZG527


docker

Current Problem the Industry is facing



A shipping container system for applications

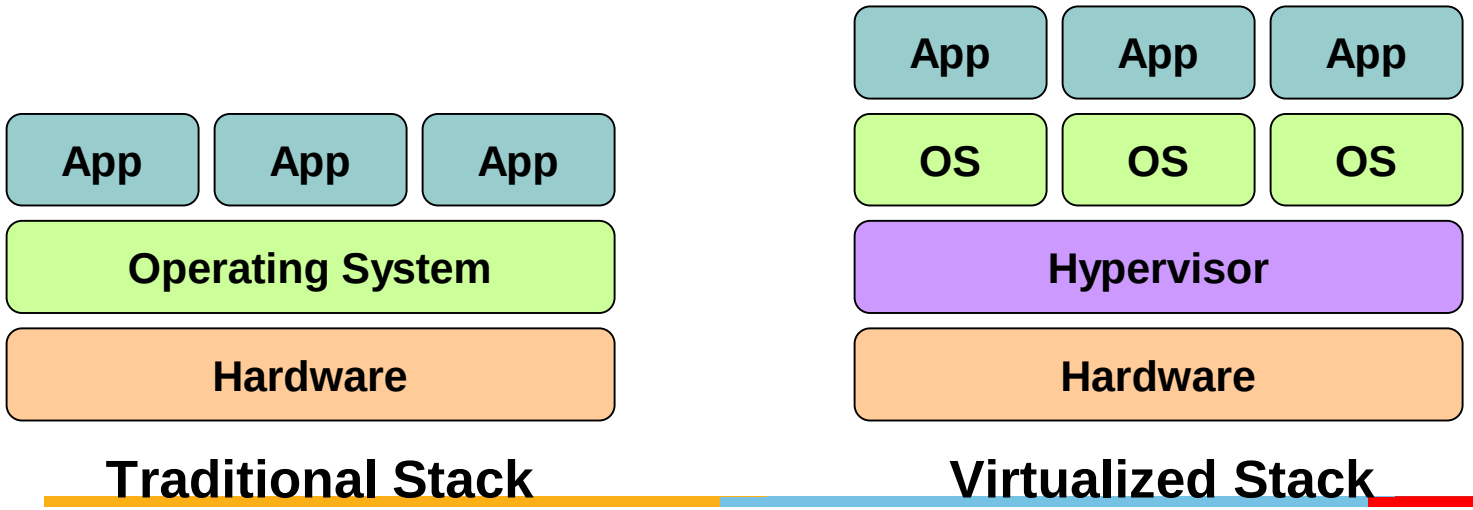


Dockers

- All applications have their own dependencies, which include both software and hardware resources.
- Docker is a mechanism that helps in isolating the dependencies per each application by packing them into containers.
- In terms of technology, it provides cloud portability by running the same applications in different virtual environments.
- Containers are scalable and safer to use and deploy as compared to regular approaches.

Virtual Machines

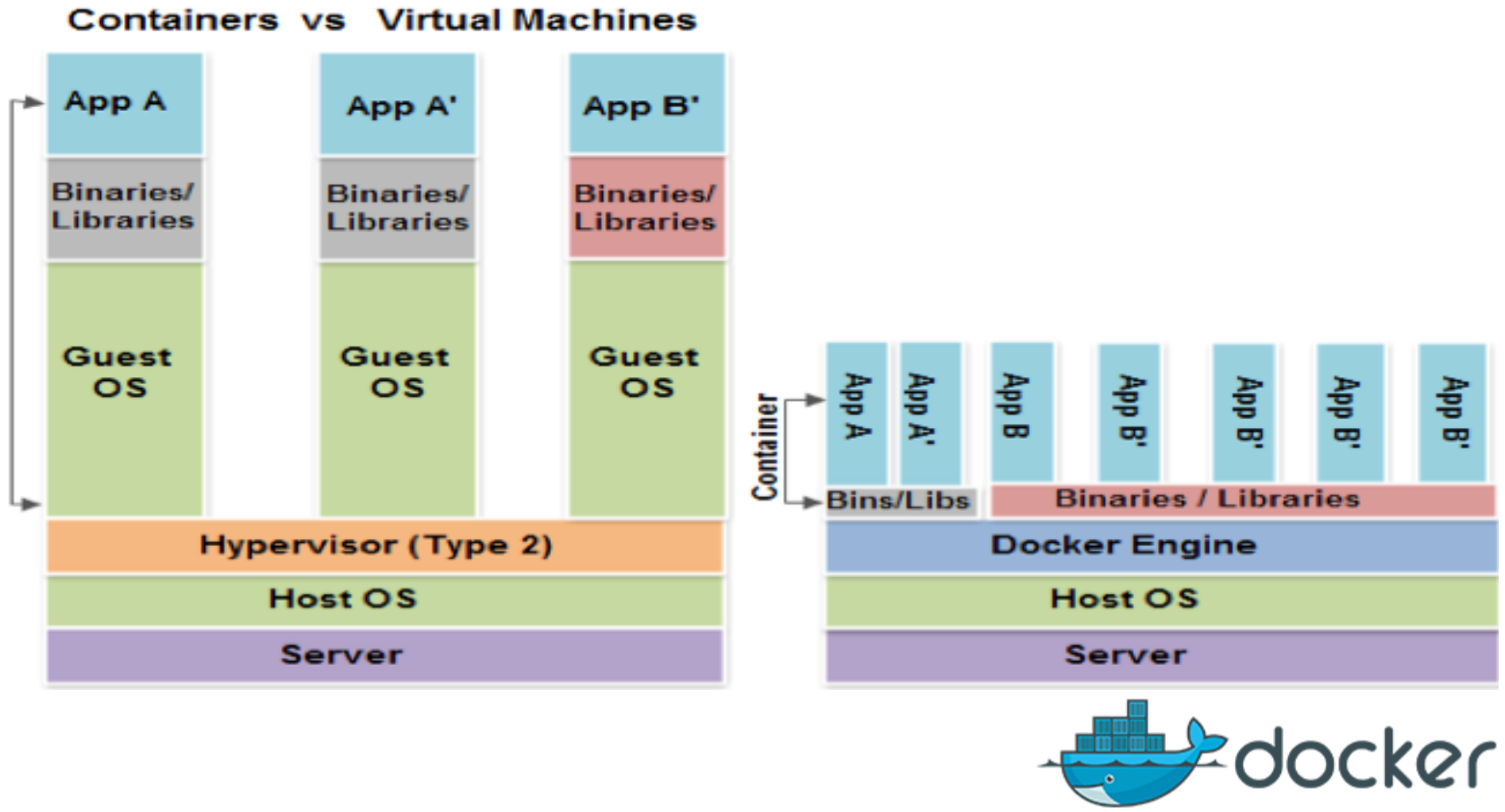
- Virtual machines are used extensively in cloud computing.
- Isolation and resource control have continually been achieved through the use of virtual machines.
- Virtual machine loads a full OS with its own memory management and enable applications to be more efficient and secure while ensuring their high availability.



How are Docker Containers different from a Virtual Machine?

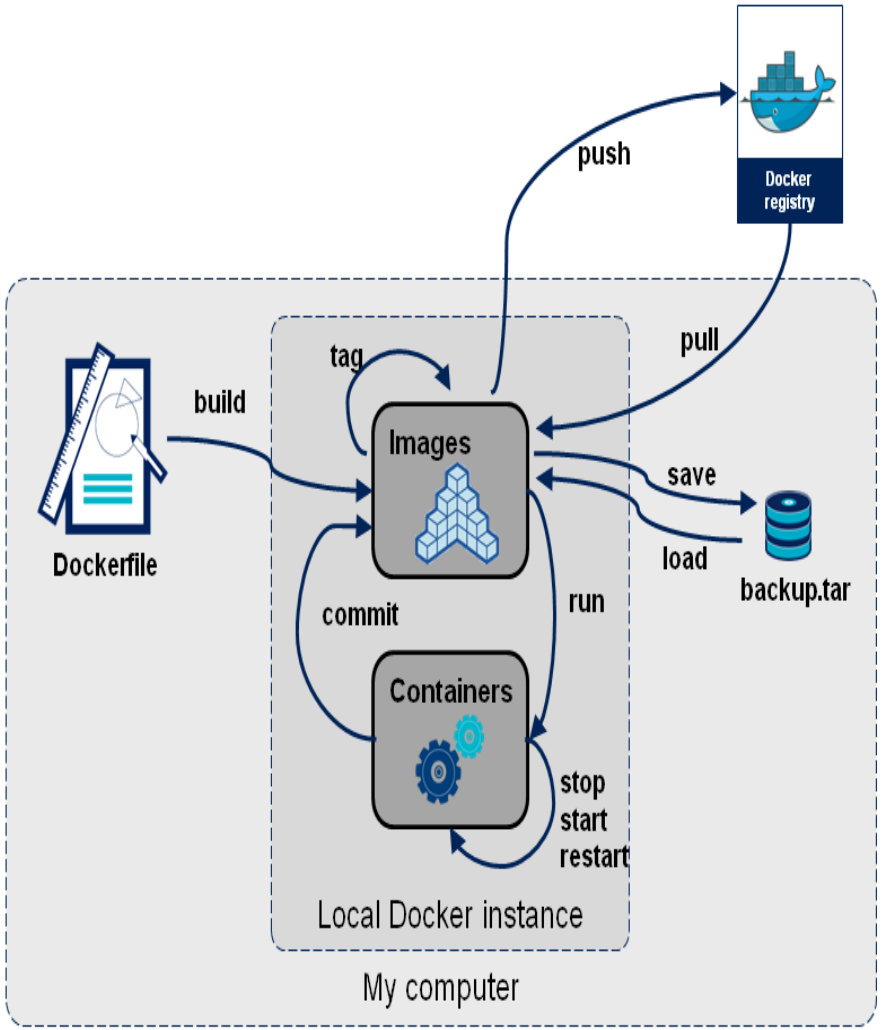
- Virtual machines have a full OS with its own memory management installed with the associated overhead of virtual device drivers.
- Docker containers are executed with the Docker engine rather than the hypervisor.
- Containers are therefore smaller than Virtual Machines and enable faster start up with better performance, less isolation and greater compatibility possible due to sharing of the host's kernel.

How are Docker Containers different from a Virtual Machine?



Docker Container Lifecycle ……

- The Life of a Container
  - Conception
    - **BUILD** an Image from a Dockerfile
  - Birth
    - **RUN** (create+start) a container
  - Reproduction
    - **COMMIT** (persist) a container to a new image
    - **RUN** a new container from an image
  - Sleep
    - **KILL** a running container
  - Wake
    - **START** a stopped container
  - Death
    - **RM** (delete) a stopped container
- Extinction
  - **RMI** a container image (delete image)



Dockerfile ……

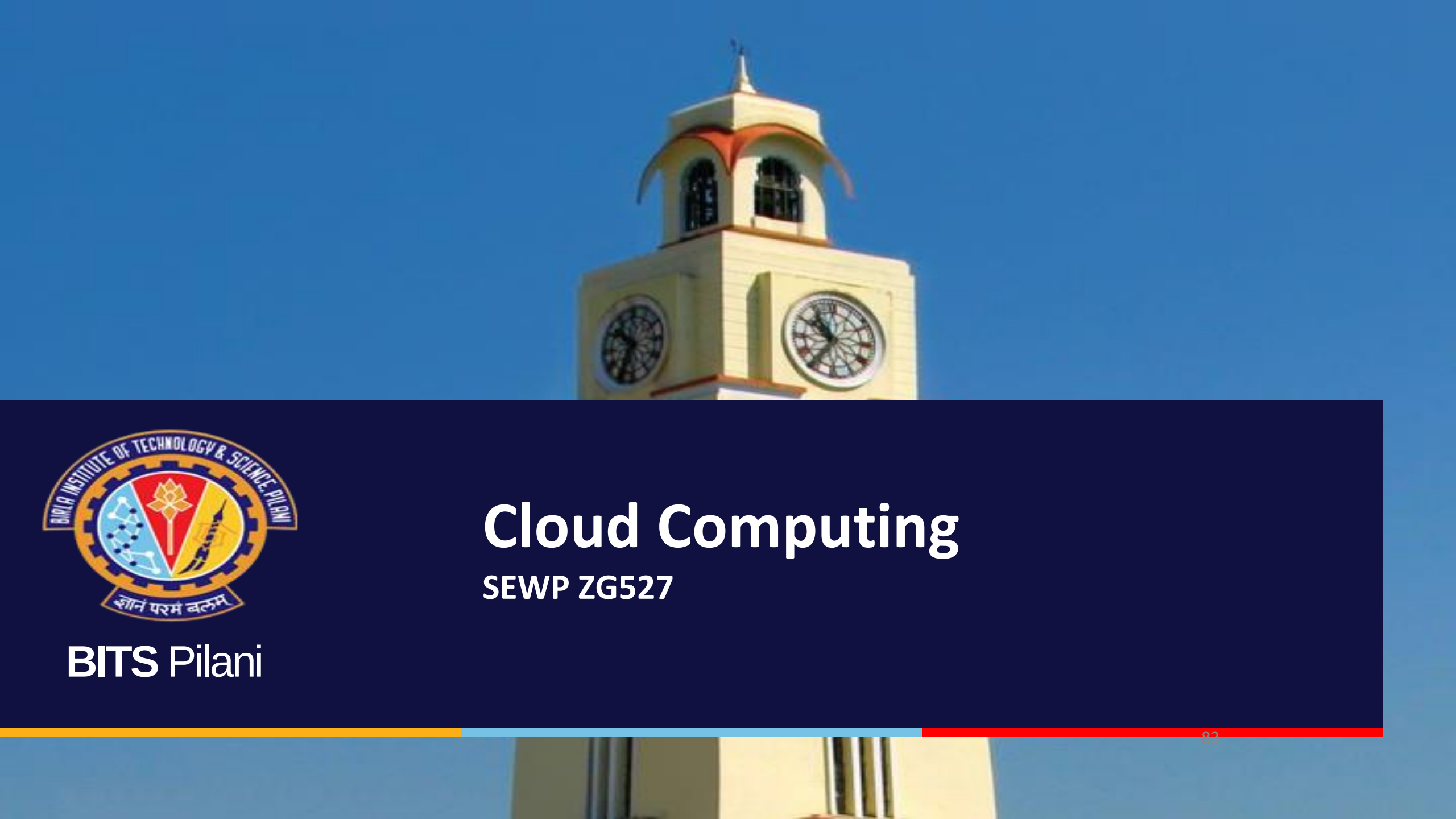
- Like a Makefile (shell script with keywords)
- Extends from a Base Image
- Results in a new Docker Image
- Imperative, not Declarative
- A Docker file lists the steps needed to build an images
- docker build is used to run a Docker file
- Can define default command for docker run, ports to expose, etc

file | 15 lines (11 sloc) | 0.475 kb | Open | Edit | Raw | Blame | History | Delete

```
1  FROM ubuntu:12.04
2
3  RUN apt-get update
4
5  # Make it easy to install PPA sources
6  RUN apt-get install -y python-software-properties
7
8  # Install Oracle's Java (Recommended for Hadoop)
9  # Auto-accept the license
10 RUN add-apt-repository -y ppa:webupd8team/java
11 RUN apt-get update
12 RUN echo oracle-java7-installer shared/accepted-oracle-license-v1-1 select true | sudo /usr/bi[illegible]con[illegible]set-selections
13 RUN apt-get -y install oracle-java7-installer
14 ENV JAVA_HOME /usr/lib/jvm/java-7-oracle
```

https://docs.docker.com/engine/installation/windows/

Thank you



BITS Pilani

Cloud Computing

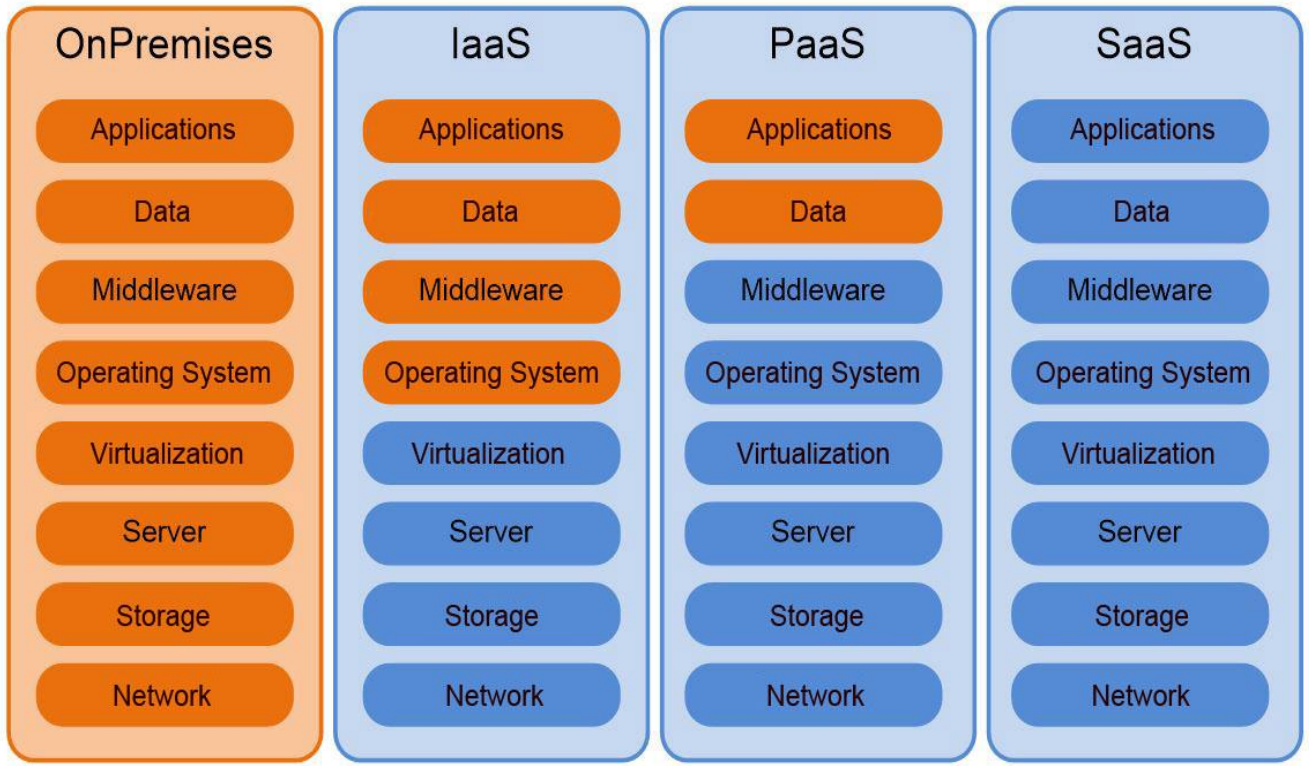
SEWP ZG527

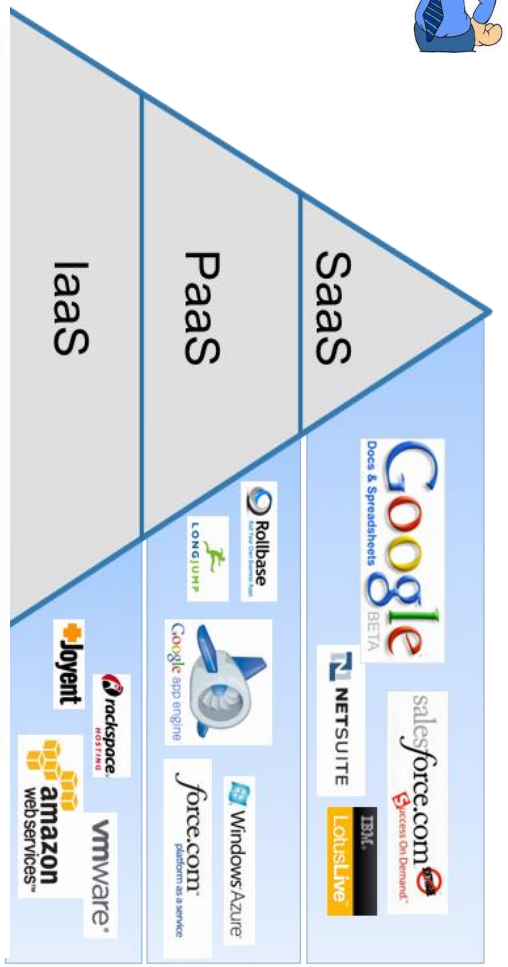
IaaS

Really, what is iaas???

heard of 3 models of Cloud Computing?

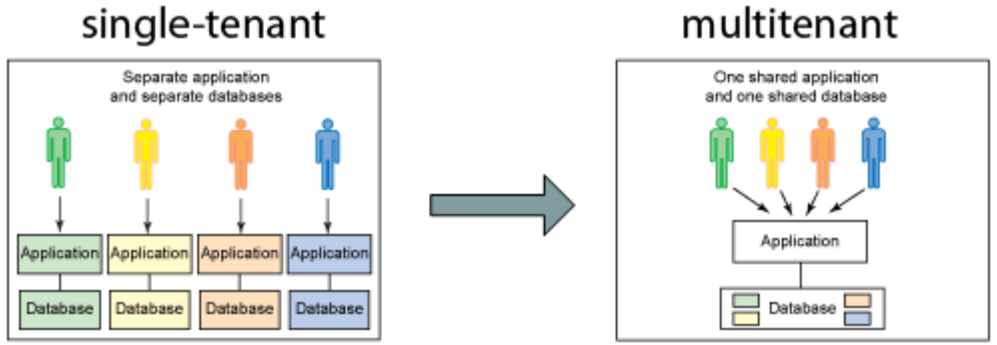
Yes, Yes, IaaS, PaaS and SaaS





Key concepts of IaaS

- Cloudbursting: The process of off-loading tasks to the cloud during times when the most compute resources are needed
- Multi-tenant computing



- Resource pooling: **Pooling** is a resource management term that refers to the grouping together of resources (compute(cpu), network(bandwidth), storage) for the purposes of ***maximizing advantage*** and/or ***minimizing risk*** to the users
- Hypervisor

Two primary facets that make IaaS special

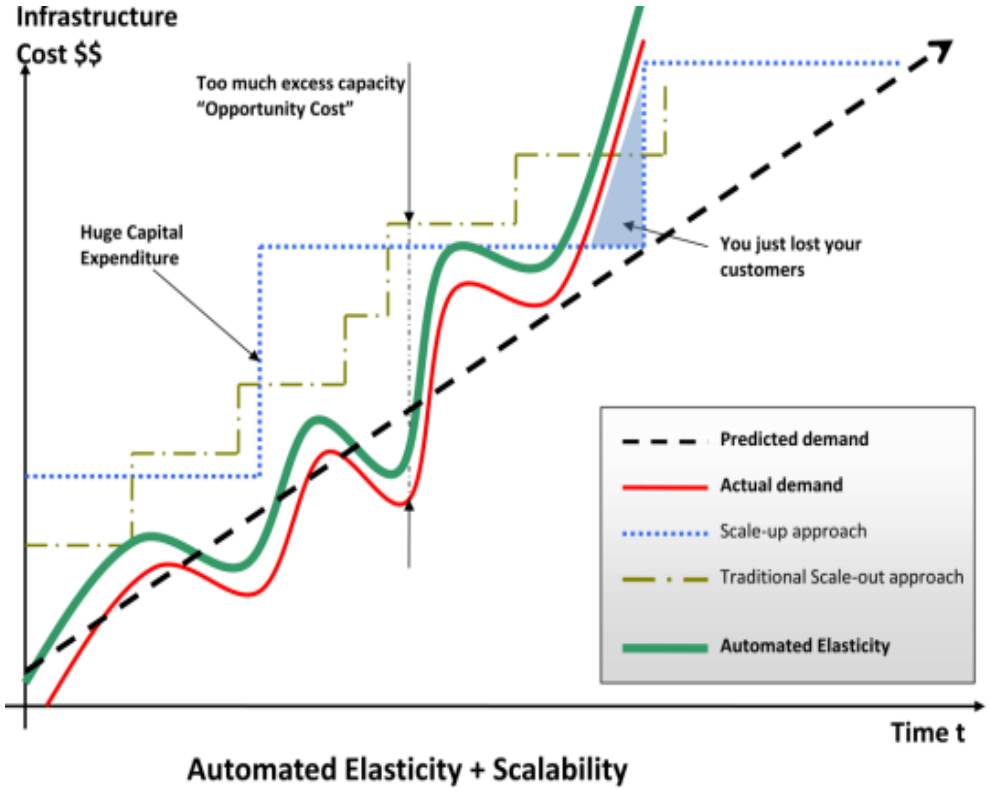
Elasticity:

Wikipedia: “In **cloud** computing, **elasticity** is defined as the degree to which a system (or a particular **cloud** layer) autonomously adapts its capacity to workload over time”

OR simply put “Ability of a system to **expand** or **contract** its dedicated resources to meet the demand”

&

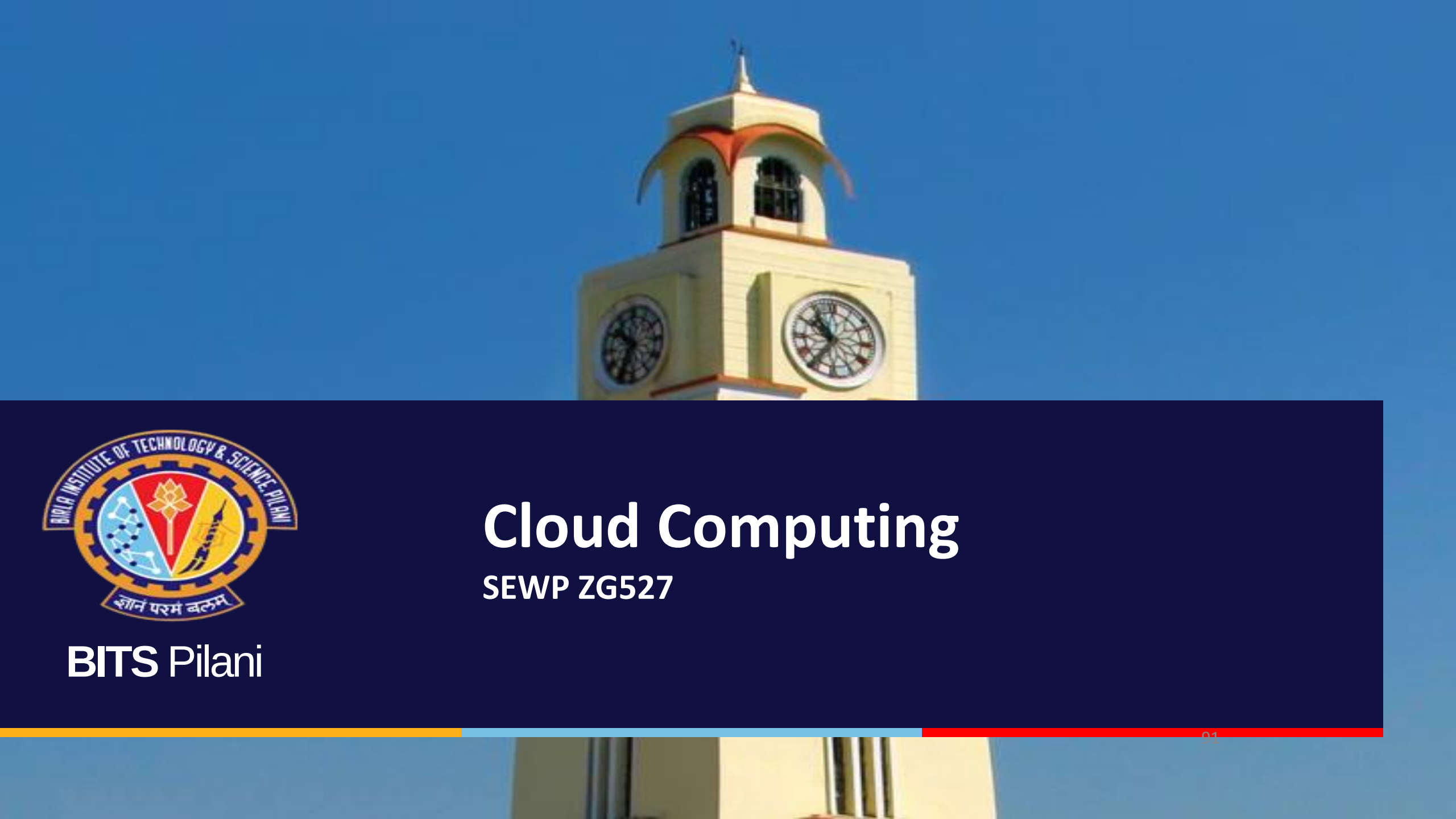
Virtualization



4 considerations:

- Developing for a specific vendor's proprietary IaaS could prove to be a costly mistake
- The complexity of well-written resource allocation software is significant and do not come cheap
- What will you be sending off to be processed in the cloud? Sending data such as personal identities, financial information, and health care data put an organization's compliance at risk
- Understand the dangers of shipping off processes that are critical to the day-to-day operation of the business.

- http://www.ibm.com/developerworks/cloud/library/cl-cloudservices1iaas/



BITS Pilani

Cloud Computing

SEWP ZG527

IaaS

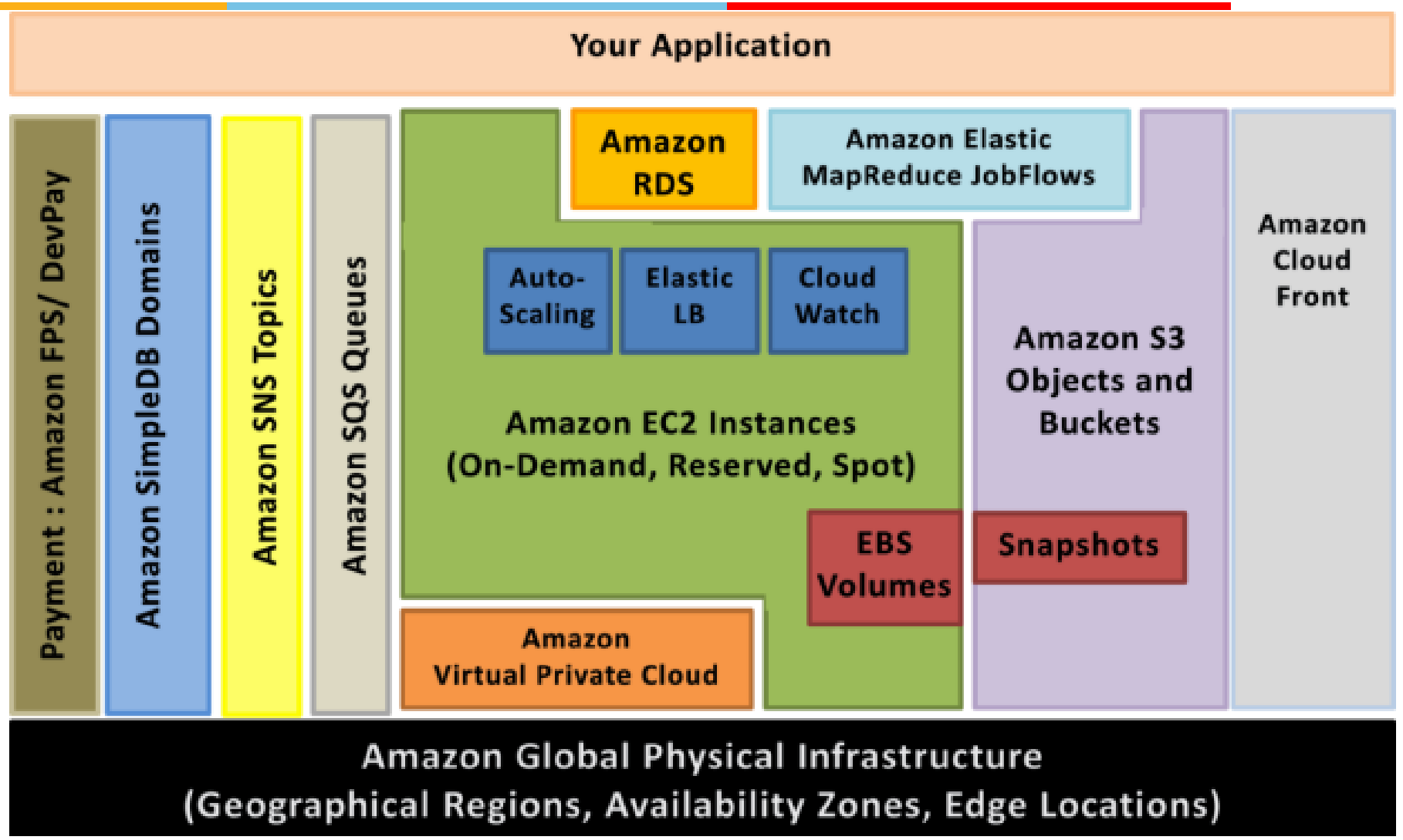
Really, what is iaas???

Amazon Web Services

Amazon Web Services Cloud

- Provides highly reliable and scalable infrastructure for deploying web-scale solutions
- With minimal support and administration costs
- More flexibility than own infrastructure, either on-premise or at a datacenter facility

AWS infrastructure services

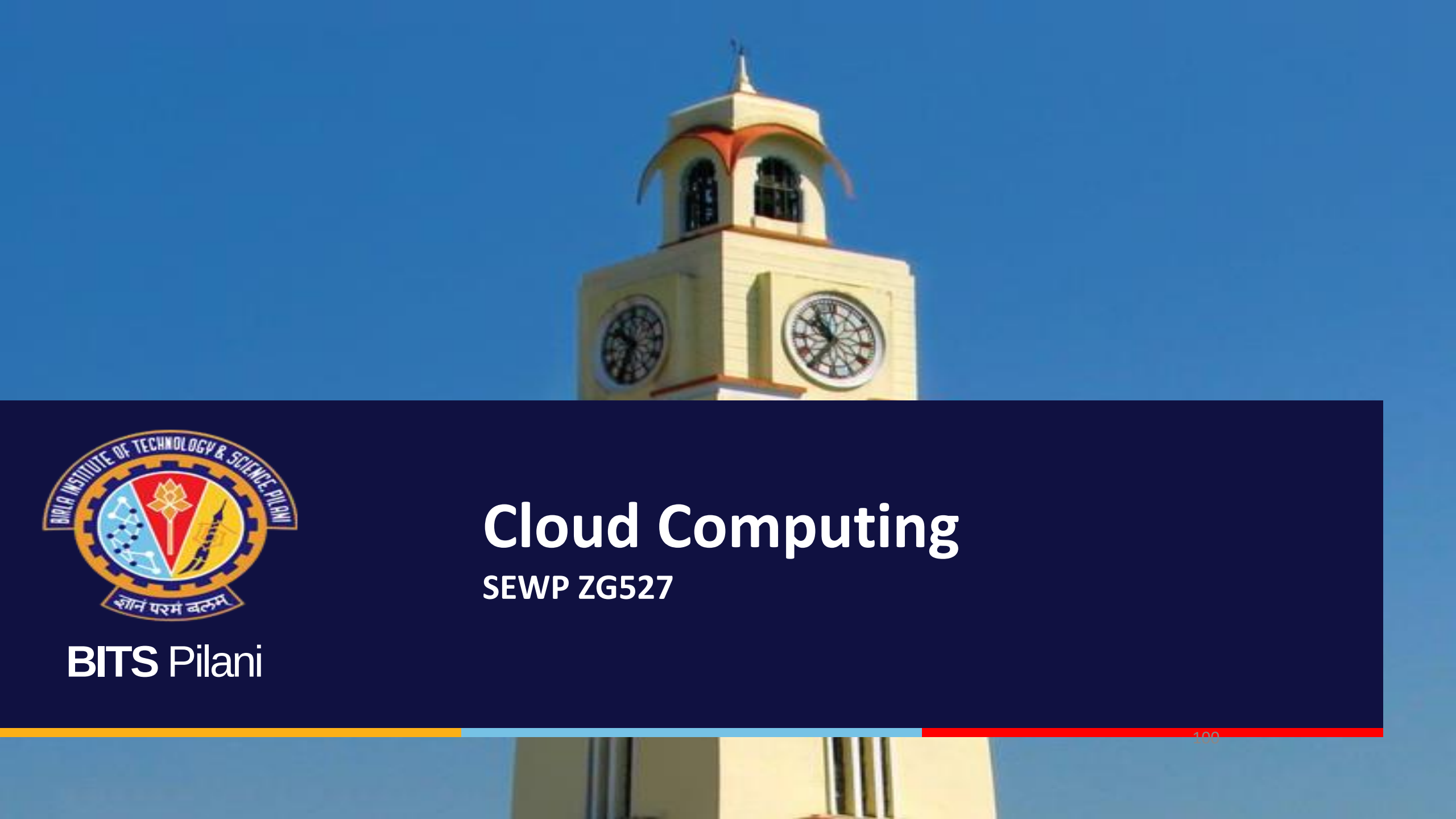


Examples

AWS – EC2, EBS, S3, LB

IaaS f  u

*Thanks, I feel so "**Clouded**" now*



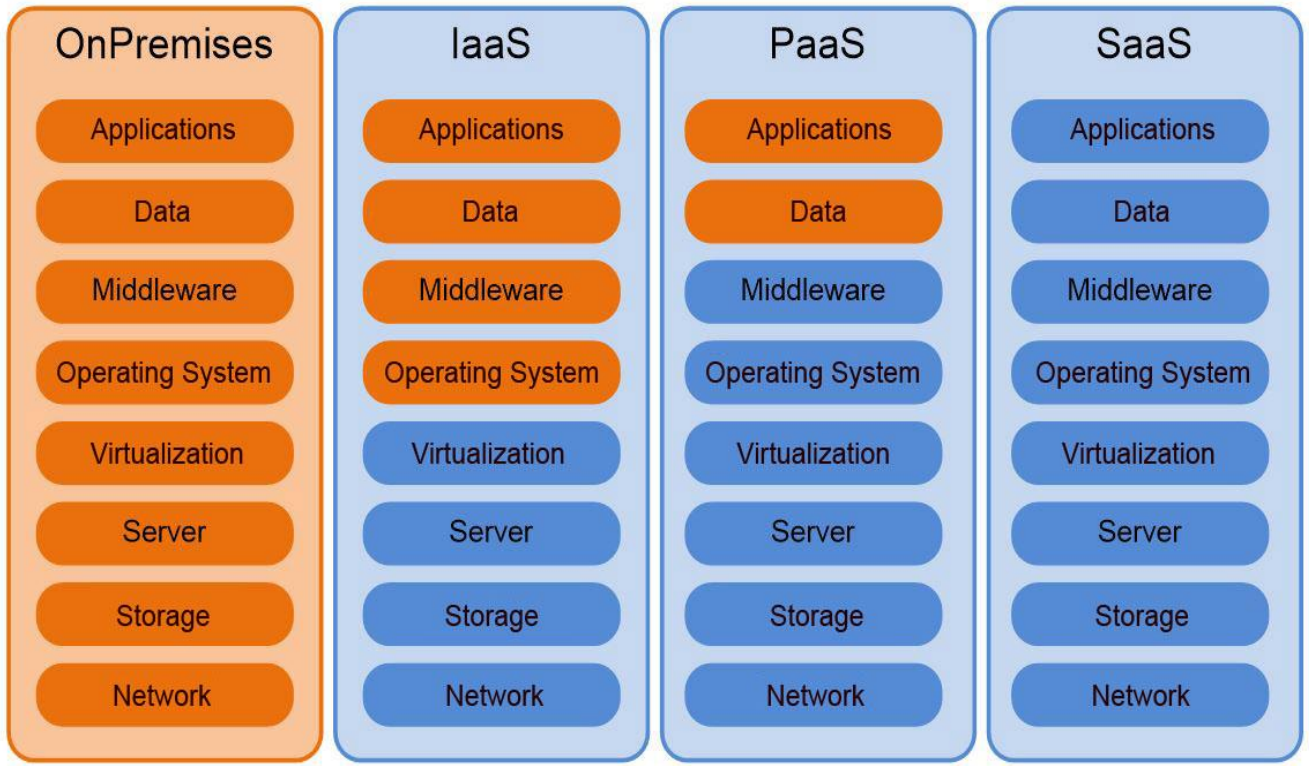
BITS Pilani

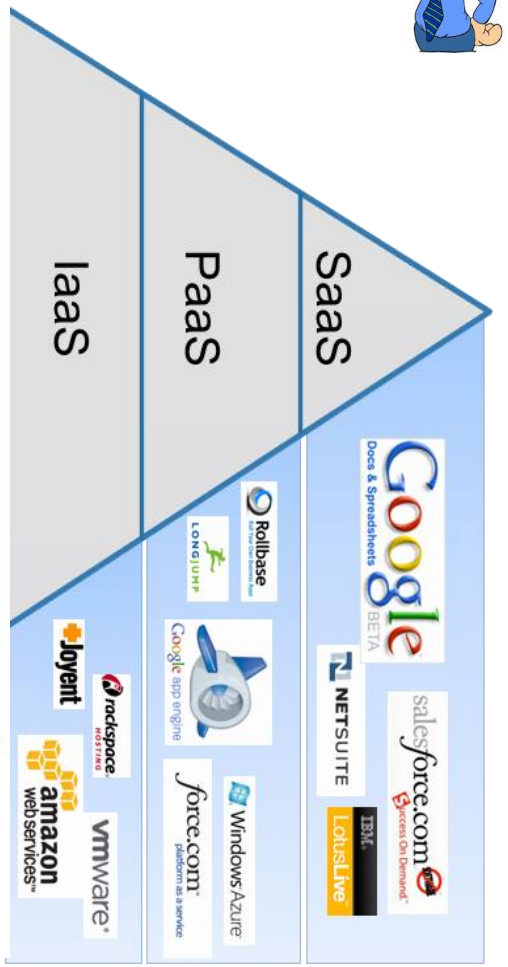
Cloud Computing

SEWP ZG527

heard of 3 models of Cloud Computing?

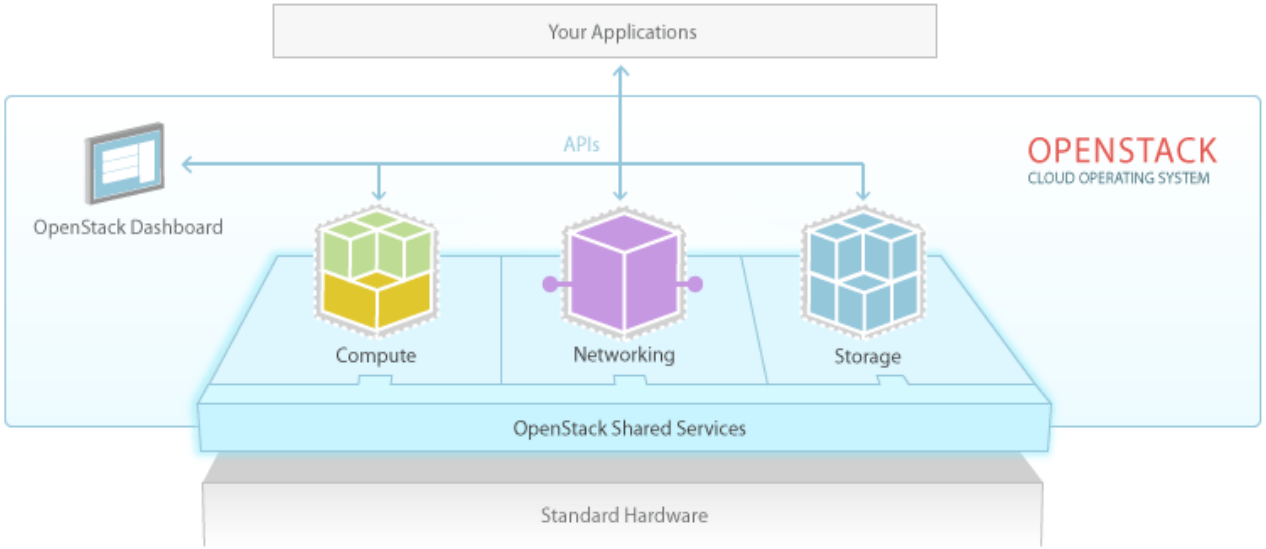
Yes, Yes, IaaS, PaaS and SaaS



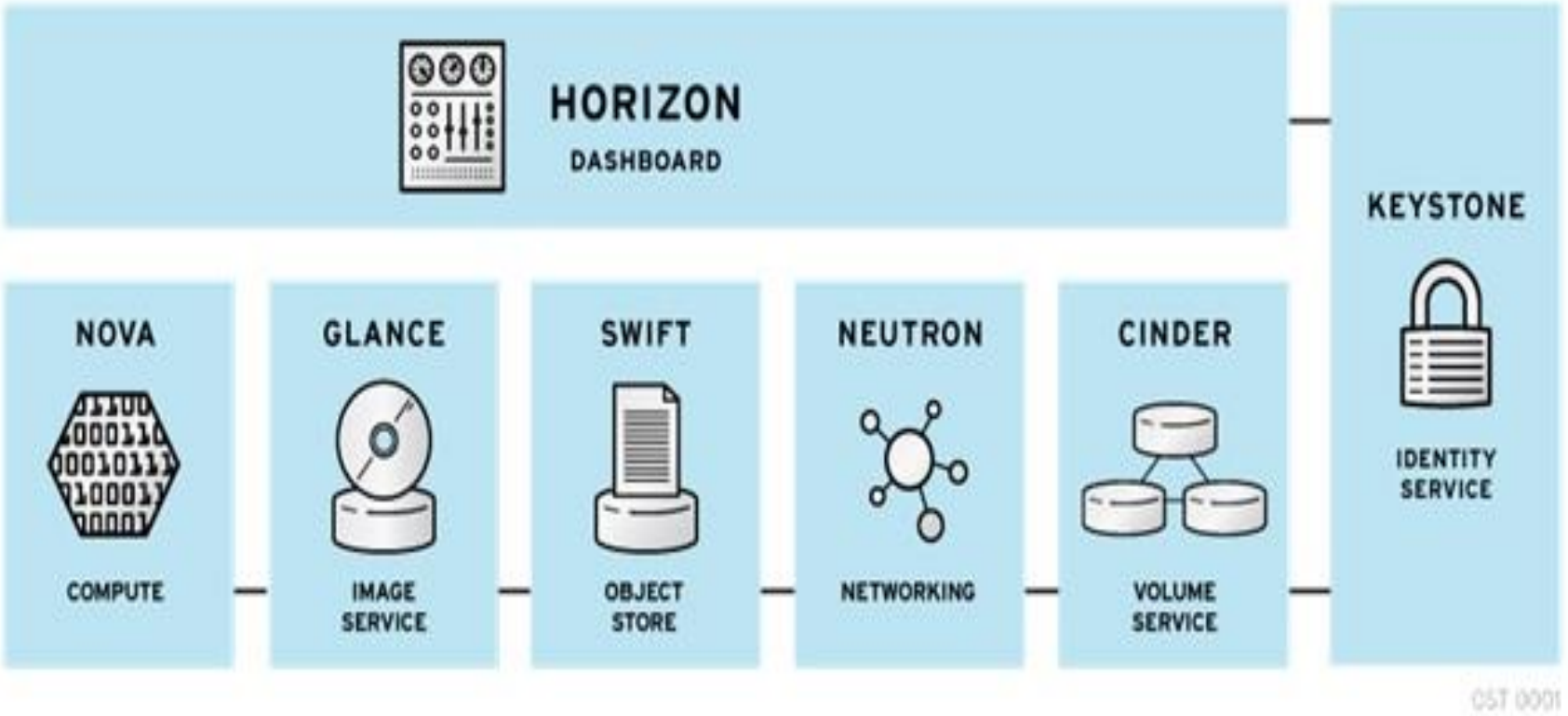


Openstack overview

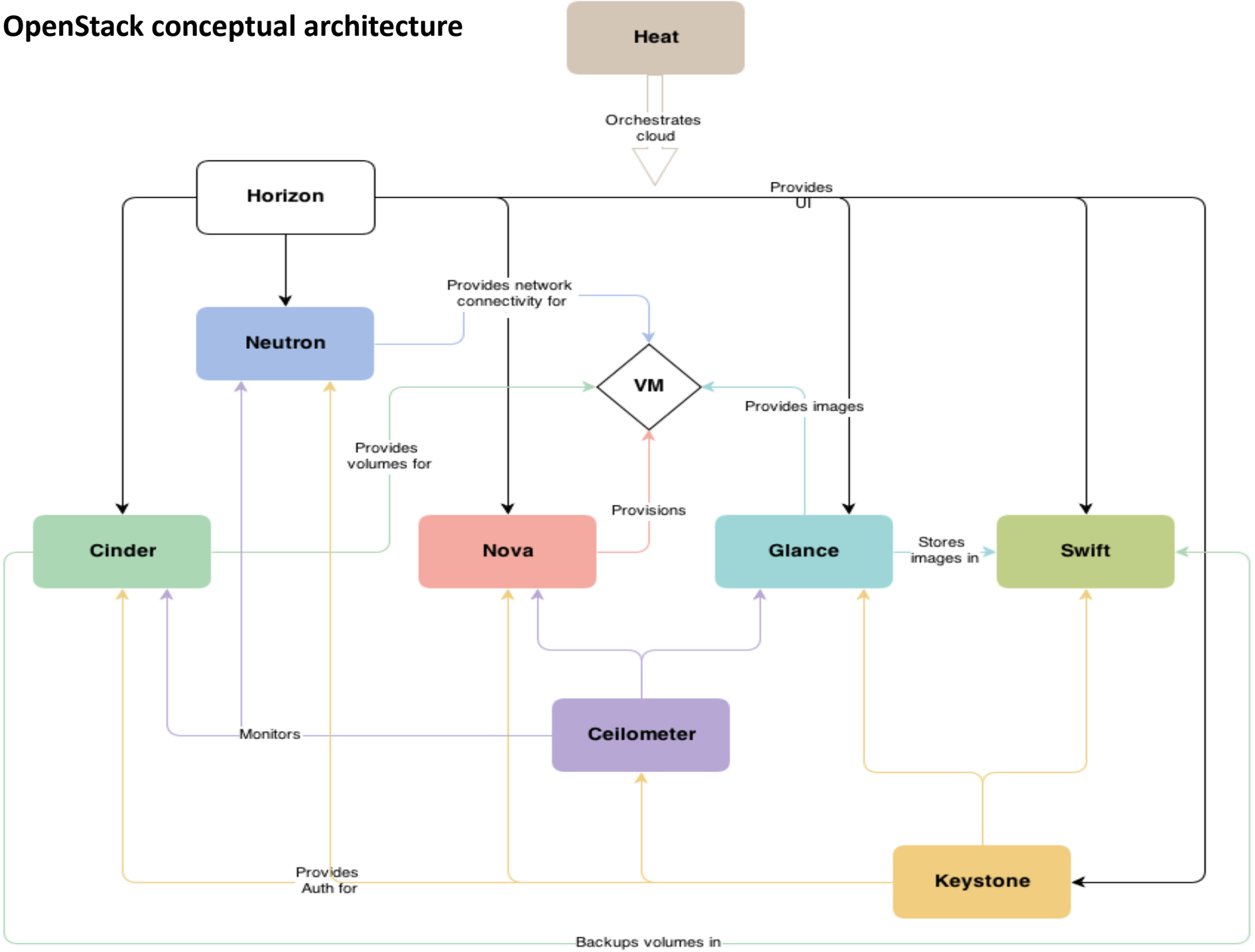
- OpenStack is a collection of open source technologies delivering a massively scalable cloud operating system.
- OpenStack cloud operating system controls large pools of compute, storage, and networking resources throughout a datacenter, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

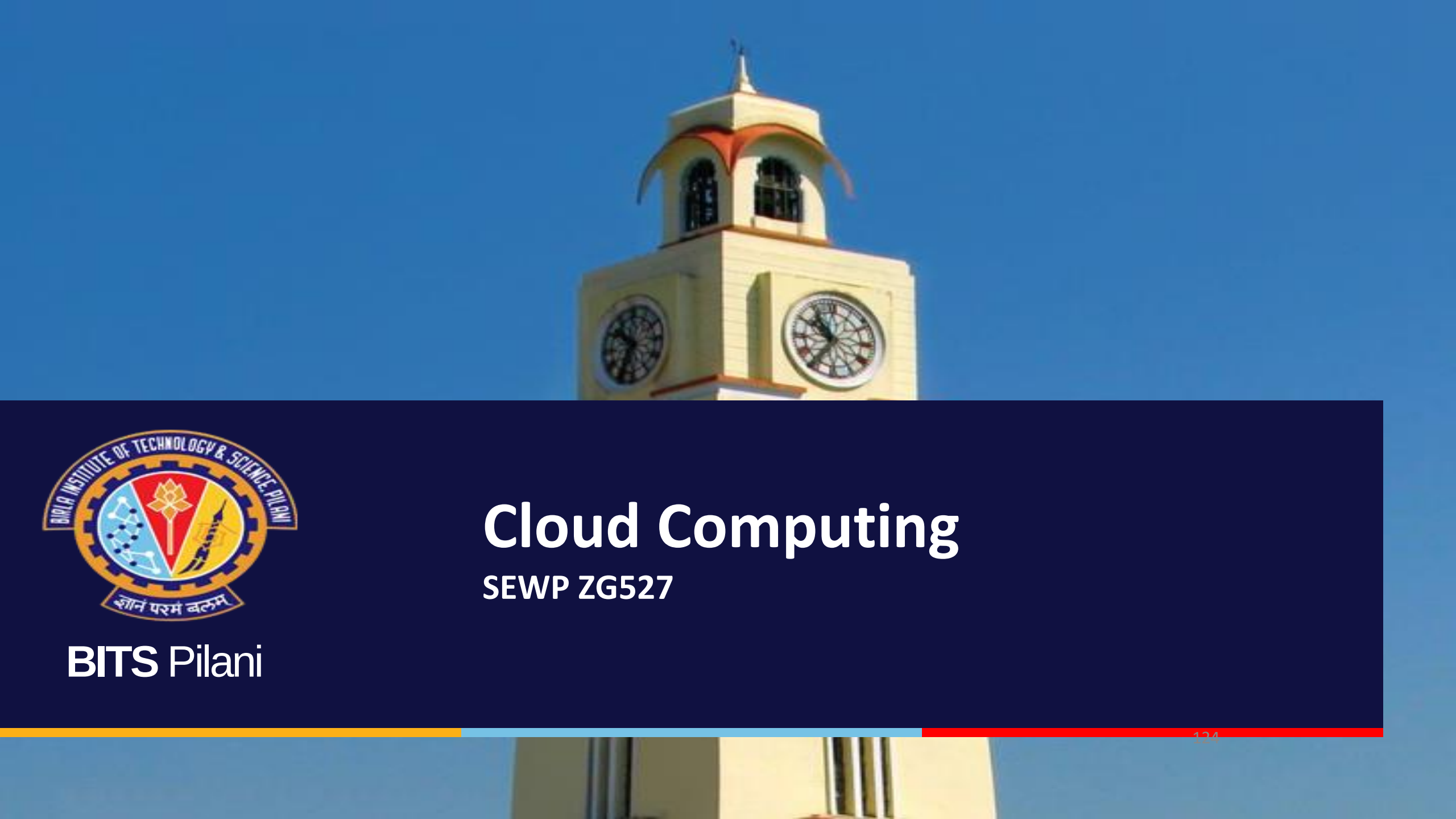


Openstack Components



OpenStack conceptual architecture





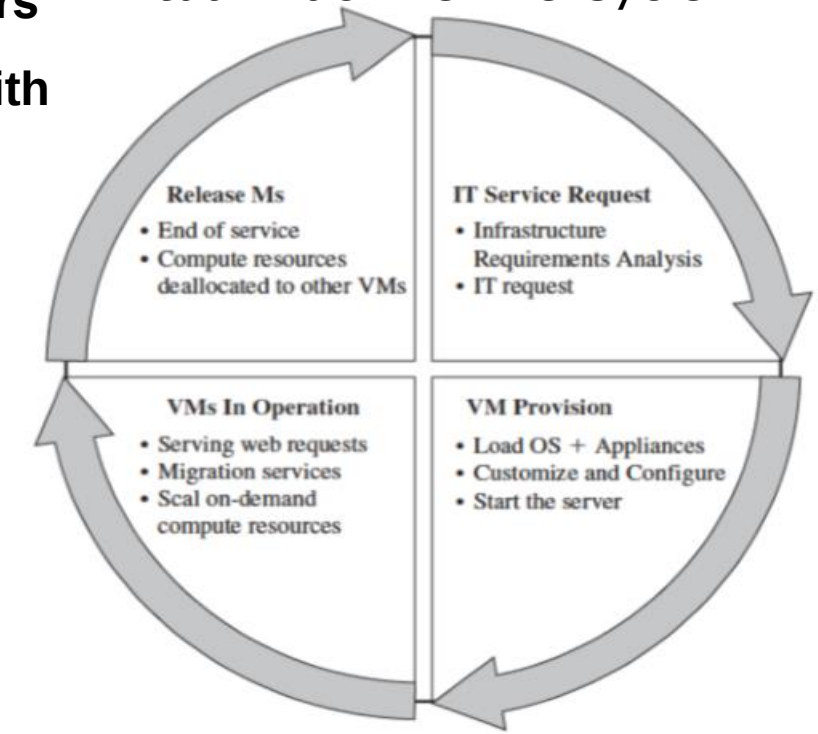
BITS Pilani

Cloud Computing

SEWP ZG527

- **The cycle starts by a request delivered to the IT department, stating the requirement for creating a new server for a particular service.**
- **This request is being processed by the IT administration to start seeing the servers' resource pool, matching these resources with requirements**
- **Starting the provision of the needed virtual machine.**
- **Once it provisioned and started, it is ready to provide the required service according to an SLA(Service Level agreement).**
- **Virtual is being released; and free resources.**

Virtual Machine Life Cycle



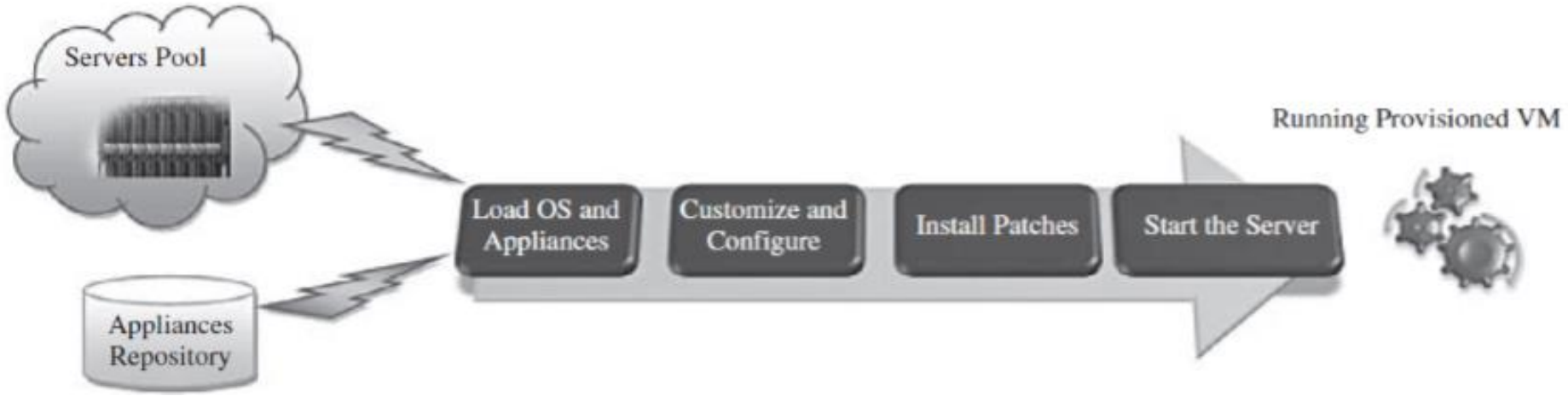
Steps to Provision VM -

• Select a server from a pool of available servers along with the appropriate OS template you need to provision the virtual machine.

• Load the appropriate software.

• Customize and configure the machine (e.g., IP address, Gateway) to an associated network and storage resources.

• Finally, the virtual server is ready to start with its newly loaded S/W.



VM Provisioning

• Server provisioning is defining server's configuration based on the organization requirements, a H/W, and S/W component (processor, RAM, storage, networking, operating system, applications, etc.).

VMs can be provisioned by

- Manually installing an OS,
- Using a preconfigured VM template,
- Cloning an existing VM, or importing a physical server or a
- Server from another hosting platform.
- Physical servers can also be virtualized and provisioned using P2V (Physical to Virtual)

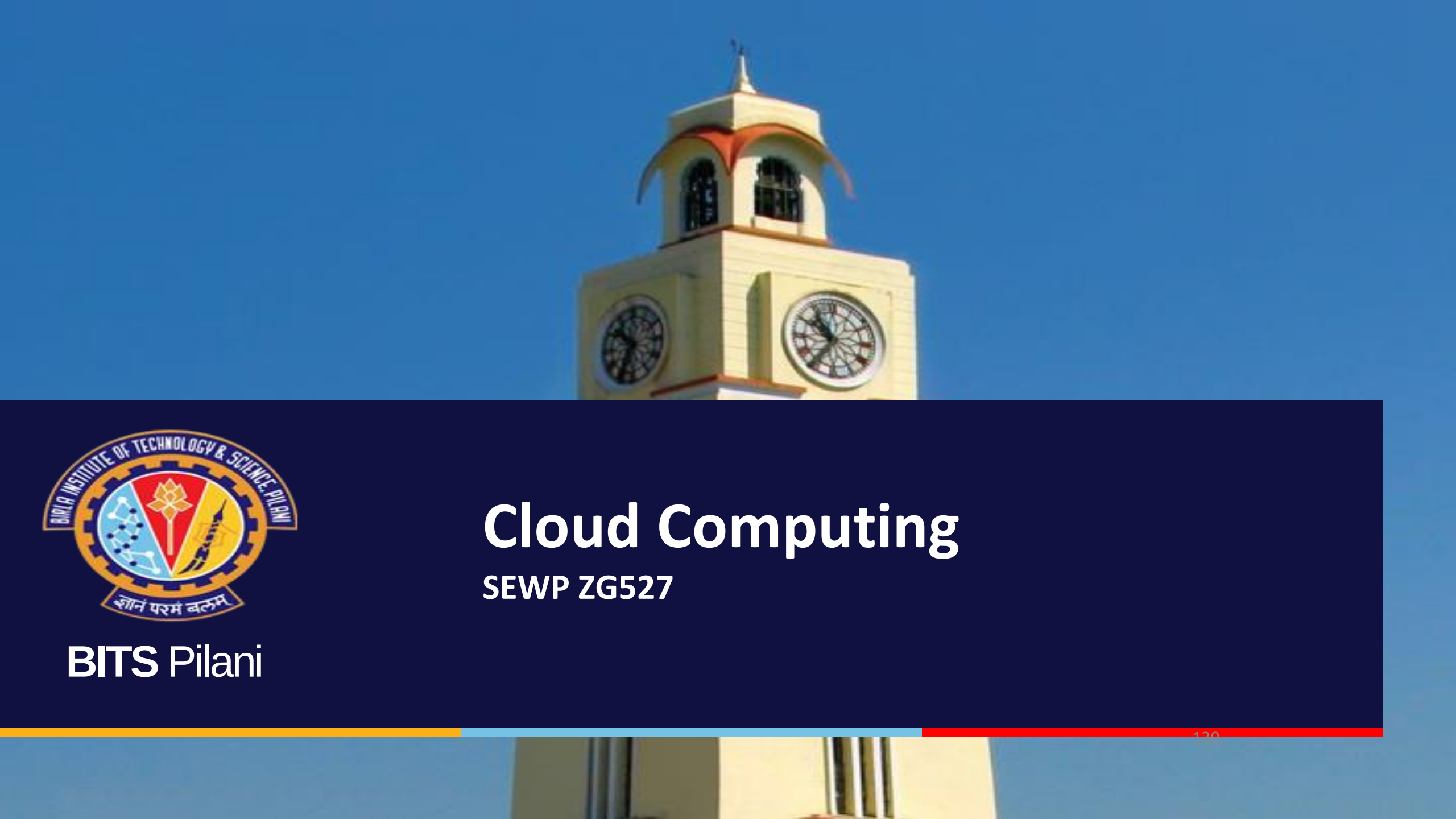
VM Provisioning using templates

- Provisioning from a template reduces the time required to create a new virtual machine.
- Administrators can create different templates for different purposes.

  For example –

  - Vagrant provision tool using VagrantFile (template file) (demo)
  - Heat – Orchestration Tool of openstack (Heat template in YAML format) (demo – Instance creation in cloud, Load balancer in cloud)

This enables the administrator to quickly provision a correctly configured virtual server on demand.



BITS Pilani

Cloud Computing

SEWP ZG527

Virtual Machine Migration Services

Migration service -

The process of moving a virtual machine from one host server or storage location to another;

There are different techniques of VM migration-

- Hot/live migration,
- Cold/regular migration, and
- Live storage migration of a virtual machine.

In this process, all key machines' components, such as CPU, storage disks, networking, and memory, are completely virtualized, thereby facilitating the entire state of a virtual machine to be captured by a set of easily moved data files.

Cold/regular migration

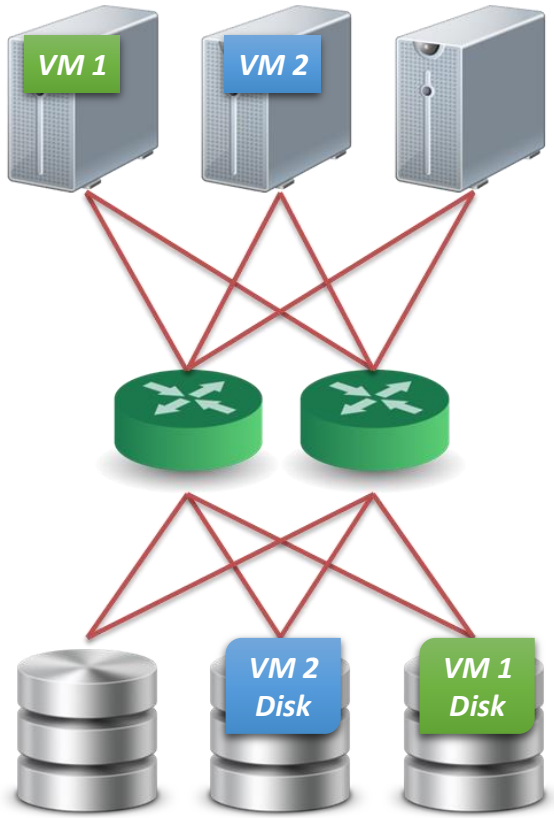
Cold migration is the migration of a powered-off virtual machine and is done in the following tasks:

- If the option to move to a different datastore was chosen, the configuration files, including the NVRAM file (BIOS settings), and log files are moved from the source host to the destination host’s associated storage area. If you chose to move the virtual machine's disks, these are also moved.
- The virtual machine is registered with the new host.
- After the migration is completed, the old version of the virtual machine is deleted from the source host if the option to move to a different datastore was chosen.

Live Migration Technique

Pre-assumption :

- We assume that all storage resources are separated from computing resources.
- Storage devices of VMs are attached from network :
  - ***NAS***: NFS, CIFS
  - ***SAN***: Fibre Channel
  - ***iSCSI***, network block device
  - ***drdb*** network RAID
- Require high quality network connection
  - Common L2 network (LAN)
  - L3 re-routing



Live Migration Technique

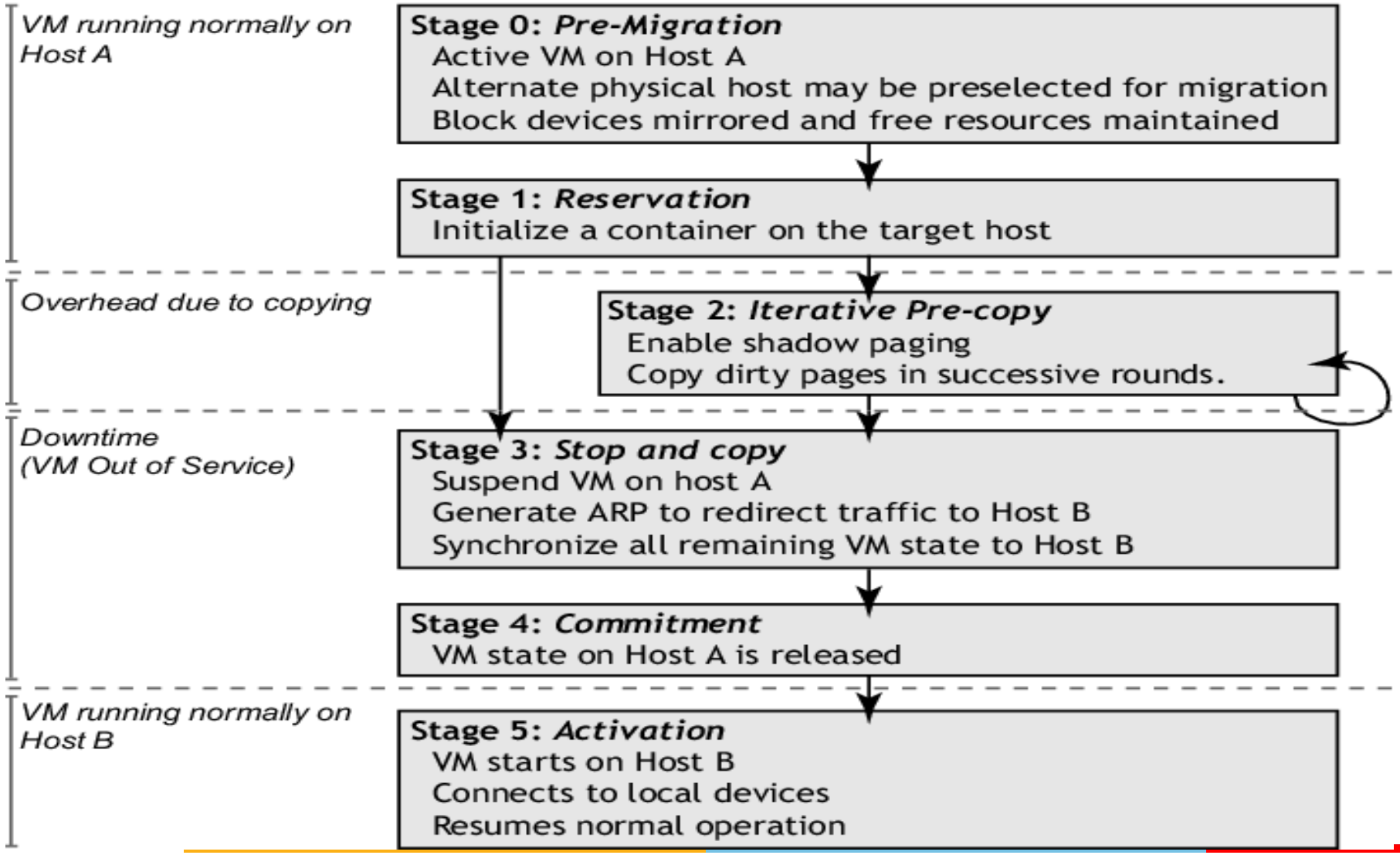
Challenges of live migration :

- VMs have lots of state in memory
- Some VMs have soft real-time requirements :
  - For examples, web servers, databases and game servers, ...etc.
  - Need to minimize down-time

Relocation strategy :

1. Pre-migration process
2. Reservation process
3. Iterative pre-copy
4. Stop and copy
5. Commitment

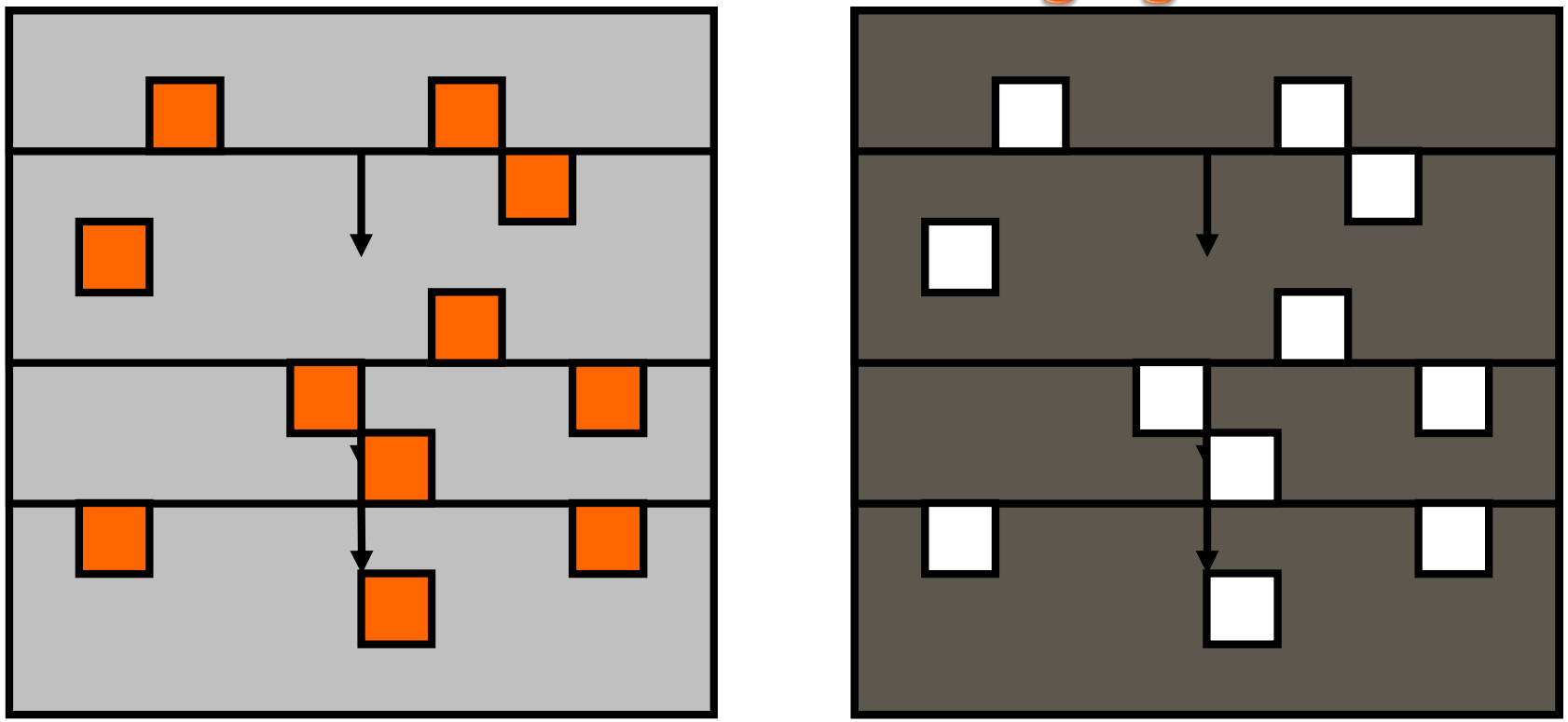
Live Migration Technique



Live Migration Technique

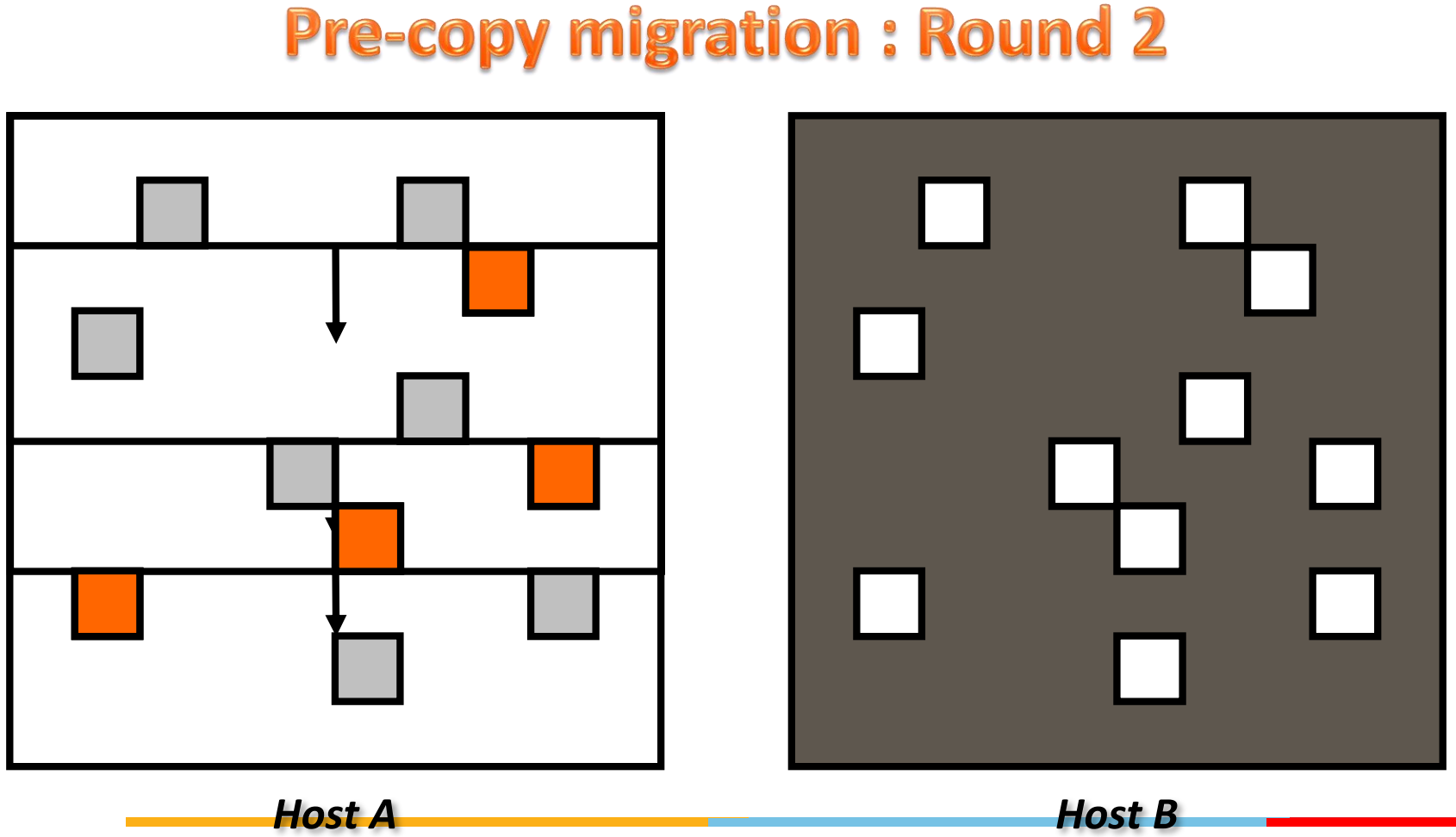
Live migration process :

Pre-copy migration : Round 1, Enable Shadow Paging



Live Migration Technique

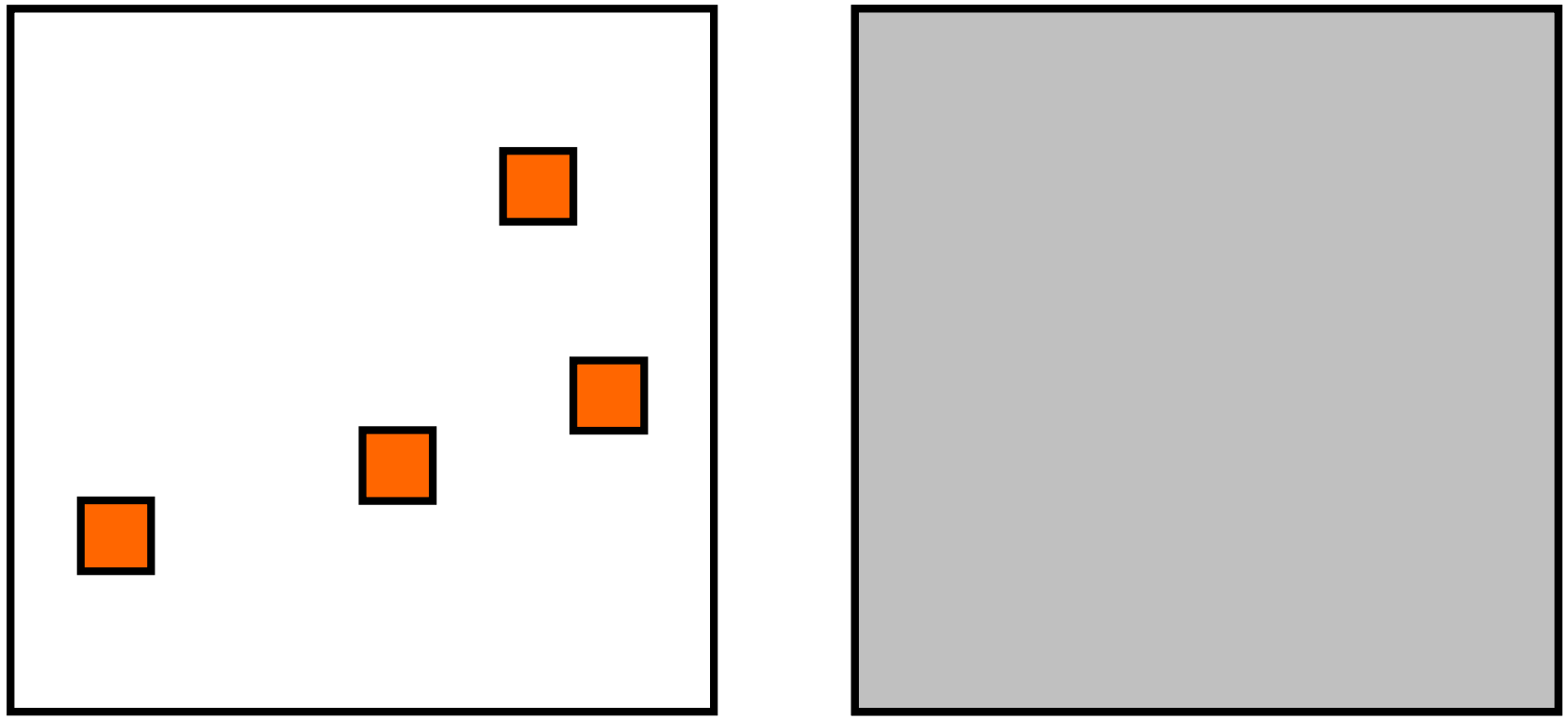
Live migration process :



Live Migration Technique

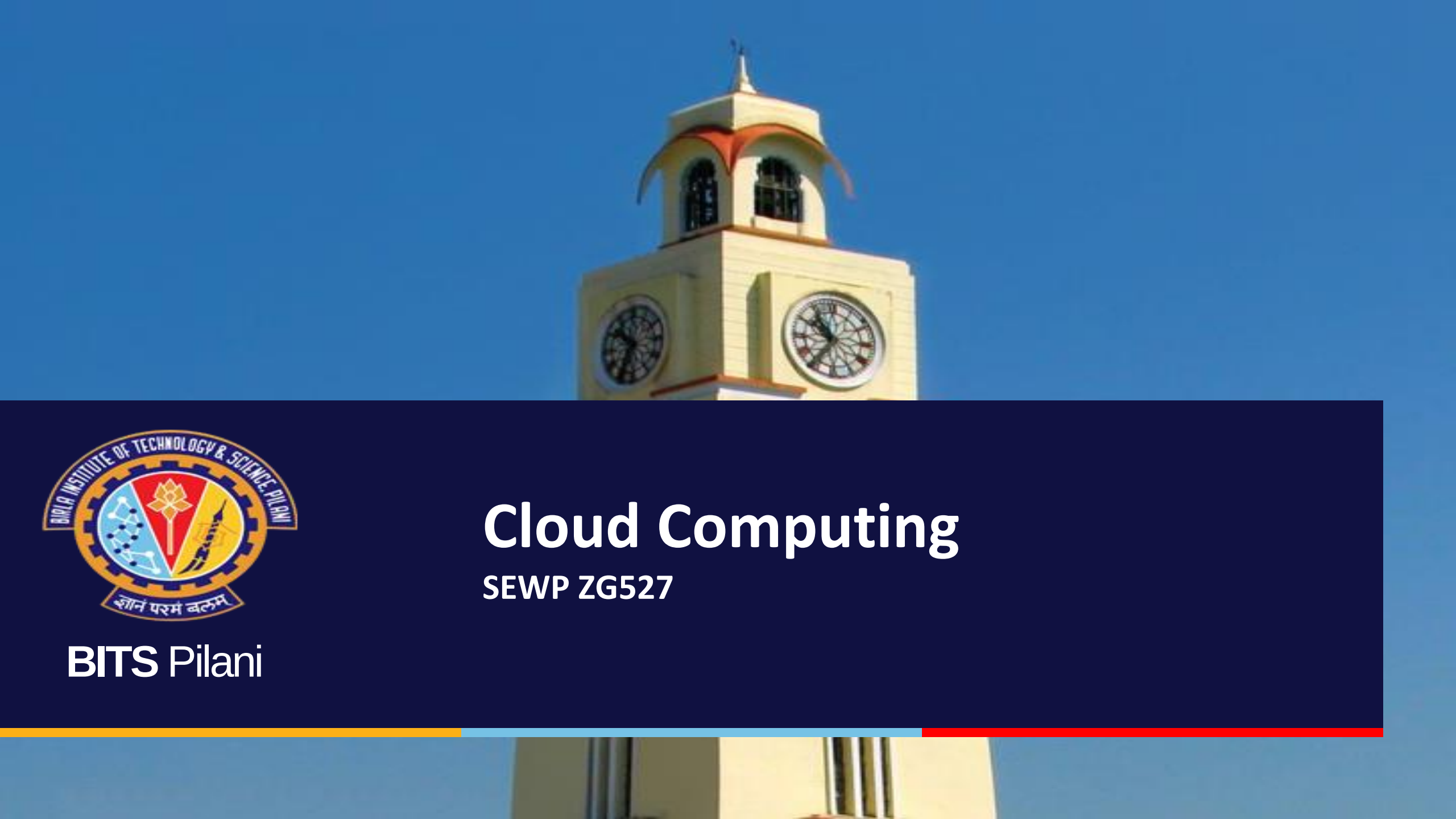
Live migration process :

Stop and copy : Final Round



Live Migration Demo

- Using Proxmox deployment tool

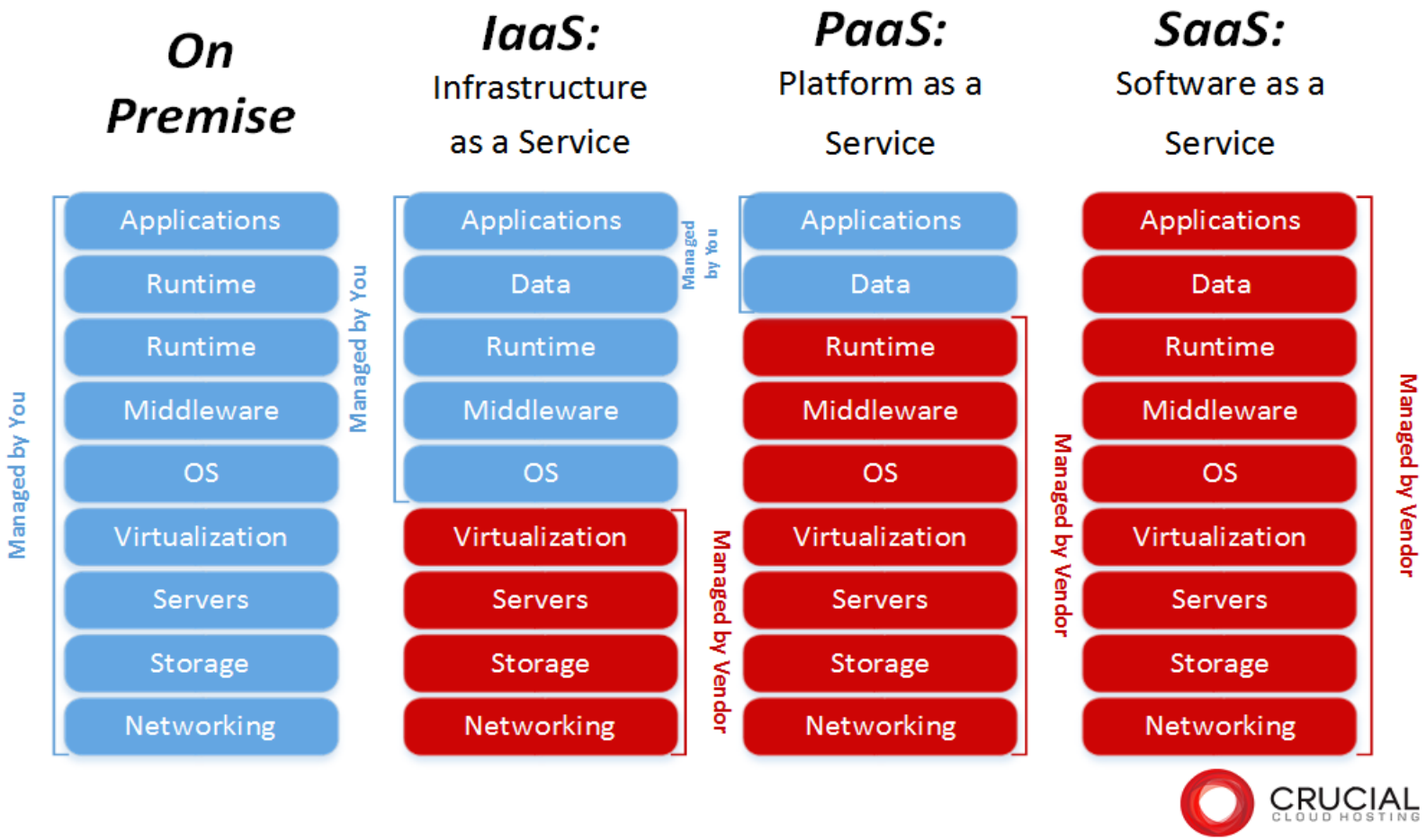


BITS Pilani

Cloud Computing

SEWP ZG527

Dependency on IaaS and PaaS

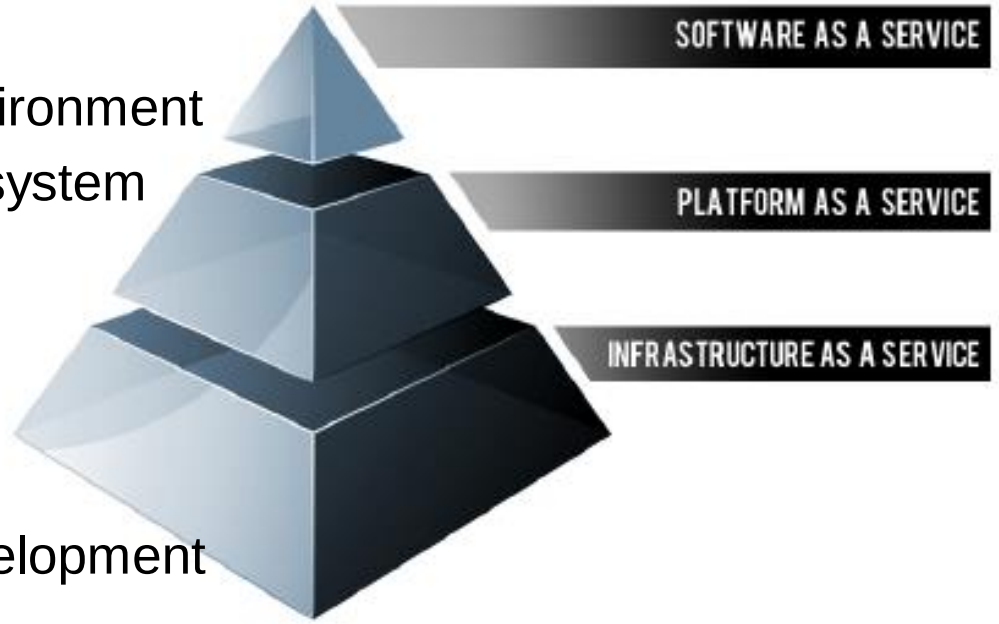


Introduction to PaaS

- Platform as a Service, referred to as PaaS, is a category of cloud computing that provides a platform and environment to allow developers to build applications and services over the internet.
- Platform as a Service allows users to create software applications using tools supplied by the provider.
- PaaS services are hosted in the cloud and accessed by users simply via their web browser.
- PaaS services can consist of preconfigured features that customers can subscribe to; they can choose to include the features that meet their requirements while discarding those that do not.

Building blocks of PaaS

- PaaS providers can assist developers from the conception of their original ideas to the creation of applications, and through to testing and deployment.
- Below are some of the features that can be included with a PaaS offering:
  - ❑ Operating system
  - ❑ Server-side scripting environment
  - ❑ Database management system
  - ❑ Server Software
  - ❑ Support
  - ❑ Storage
  - ❑ Network access
  - ❑ Tools for design and development
  - ❑ Hosting



Characteristics of PAAS

- Services to develop, test, deploy, host and maintain applications in the same integrated development environment. All the varying services needed to fulfill the application development process
- Web based user interface creation tools help to create, modify, test and deploy different UI scenarios
- Multi-tenant architecture where multiple concurrent users utilize the same development application
- Built in scalability of deployed software including load balancing and failover
- Integration with web services and databases via common standards
- Support for development team collaboration – some PaaS solutions include project planning and communication tools
- Tools to handle billing and subscription management

Characteristics of PAAS

PaaS, which is similar in many ways to Infrastructure as a Service, is differentiated from IaaS by the addition of value added services and comes in two distinct flavours;

1. A collaborative platform for software development, focused on workflow management regardless of the data source being used for the application. An example of this approach would be Heroku, a PaaS that utilizes the Ruby on Rails development language.
2. A platform that allows for the creation of software utilizing proprietary data from an application. This sort of PaaS can be seen as a method to create applications with a common data form or type. An example of this sort of platform would be the Force.com. PaaS from Salesforce.com which is used almost exclusively to develop applications that work with the Salesforce.com CRM

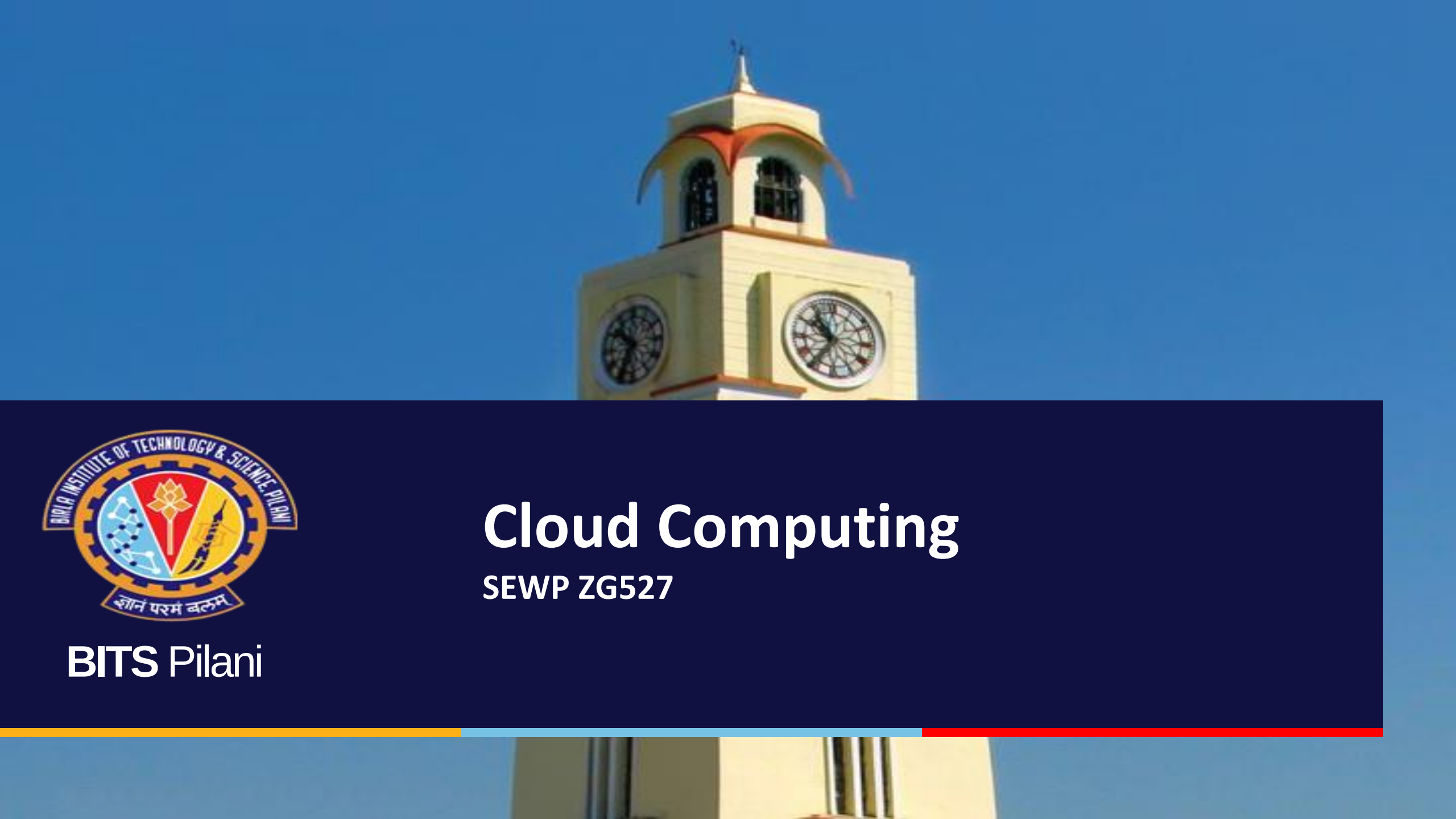
Advantages

- Users don’t have to invest in physical infrastructure
- PaaS allows developers to frequently change or upgrade operating system features. It also helps development teams collaborate on projects.
- Makes development possible for ‘non-experts’
- Teams in various locations can work together
- Security is provided, including data security and backup and recovery.
- Adaptability; Features can be changed if circumstances dictate that they should.
- Flexibility; customers can have control over the tools that are installed within their platforms and can create a platform that suits their specific requirements. They can ‘pick and choose’ the features they feel are necessary.

Advantages and Risks

Risks

- Since users rely on a provider's infrastructure and software, vendor lock-in can be an issue in PaaS environments.
- Other risks associated with PaaS are provider downtime or a provider changing its development roadmap.
- If a provider stops supporting a certain programming language, users may be forced to change their programming language, or the provider itself. Both are difficult and disruptive steps.



BITS Pilani

Cloud Computing

SEWP ZG527

Paas Example

- PaaS does not typically replace a business' entire infrastructure. Instead, a business relies on PaaS providers for key services, such as Java development or application hosting.
- For example:

  Deploying a typical business tool locally might require an IT team to buy and install hardware, operating systems, middleware (such as databases, Web servers and so on) the actual application, define user access or security, and then add the application to existing systems management or application performance monitoring (APM) tools. IT teams must then maintain all of these resources over time.

  Paas solution: A PaaS provider, however, supports all the underlying computing and software; users only need to log in and start using the platform – usually through a Web browser interface.

Paas Example: Windows Azure

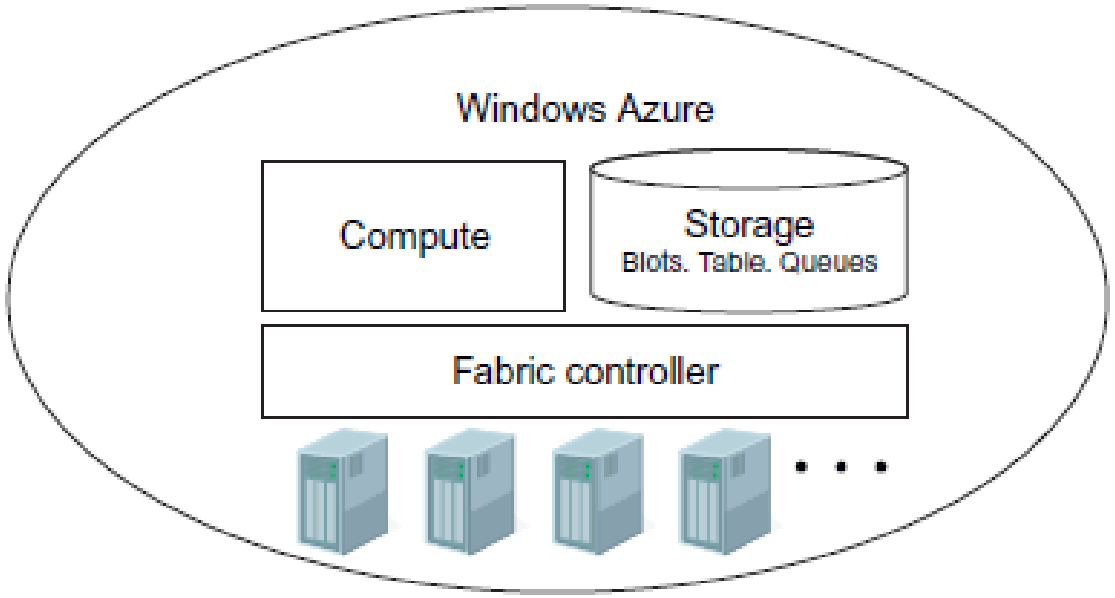
- Windows Azure is Microsoft's operating system for cloud computing.
- Windows Azure is intended to simplify IT management and minimize up-front and ongoing expenses
- To this end, Azure was designed to facilitate the management of scalable Web applications over the Internet.
- Windows Azure can be used to create, distribute and upgrade Web applications without the need to maintain expensive, often underutilized resources onsite.
- New Web services and applications can be written and debugged with a minimum of overhead and personnel expense.

Paas Example: Windows Azure

- The Azure operating system is the central component of the company's Azure Services Platform, which also includes separate application, security, storage and virtualization service layers and a desktop development environment.
- Windows Azure supports a wide variety of Microsoft and third-party standards, protocols, programming languages and platforms. Examples include XML (Extensible Markup Language), REST (representational state transfer), SOAP (Simple Object Access Protocol), Eclipse, Ruby, PHP and Python.
- Although it faces steep competition from Amazon Web Services (AWS), Microsoft Azure has managed to hold a strong second place among cloud hosting platform providers. http://azure.microsoft.com/en-us/

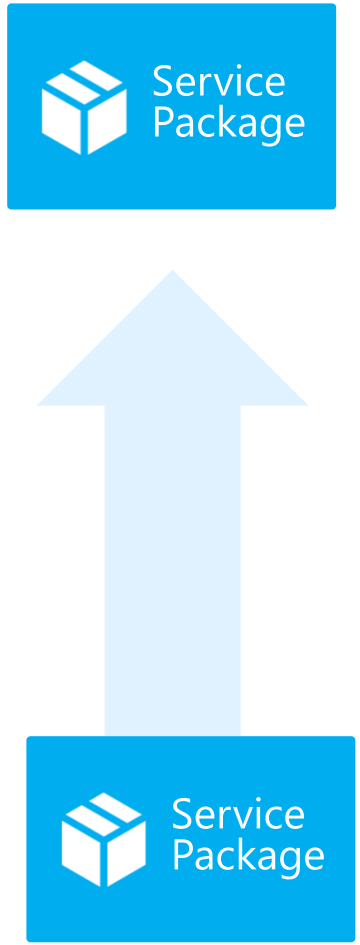
Windows Azure Runtime Environment

- The Windows Azure runtime environment provides a scalable compute and storage hosting environment along with management capabilities. It has three major components: Compute, Storage and the Fabric Controller



Windows Azure Runtime Environment

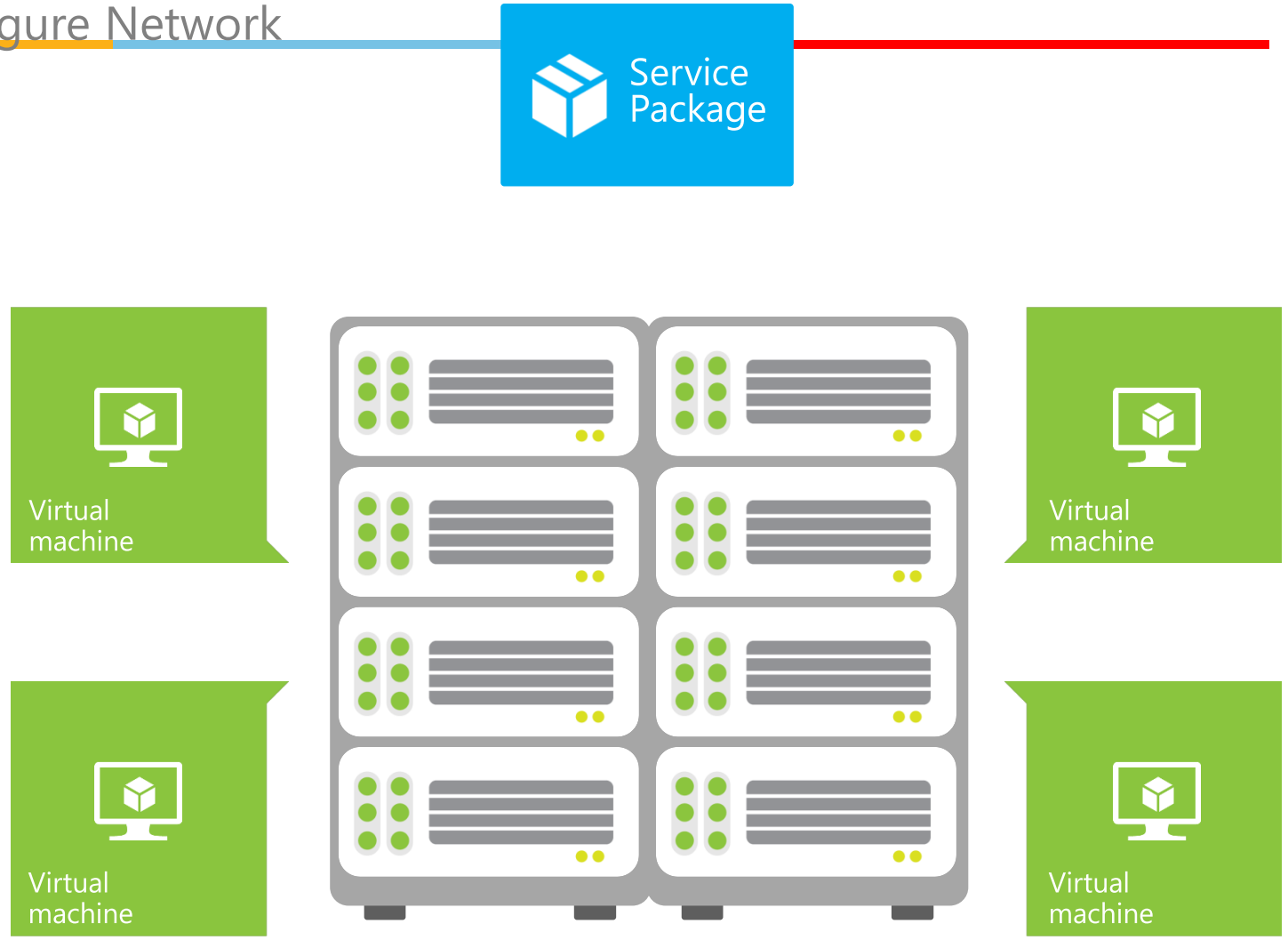
- The hosting environment of Azure is called the **Fabric Controller**. It has a pool of individual systems connected on a network and automatically manages resources by load balancing and geo-replication. It manages the application lifecycle without requiring the hosted apps to explicitly deal with the scalability and availability requirements. Each physical machine hosts an Azure agent that manages the machine.
- The **Azure Compute Service** provides a Windows-based environment to run applications written in the various languages and technologies supported on the Windows platform.
- The Windows **Azure storage service** provides scalable storage for applications running on the Windows Azure in multiple forms. It enables storage for binary and text data, messages and structured data through support for features called Blobs,Tables, Queues and Drives.


Service Package
Service Package

Provision Role Instances

Deploy App Code

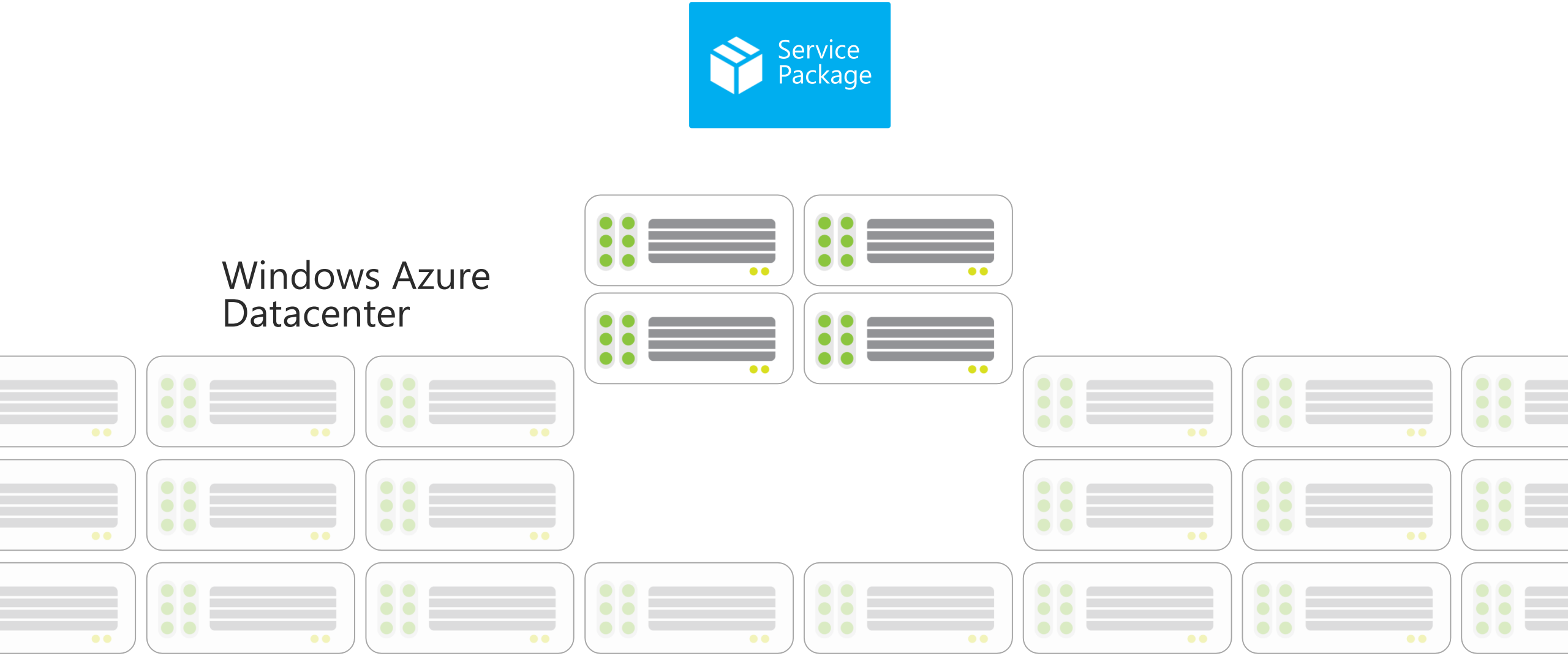
Configure Network



Provision Role Instances

Deploy App Code

Configure Network



Provision Role Instances

Deploy App Code

Configure Network

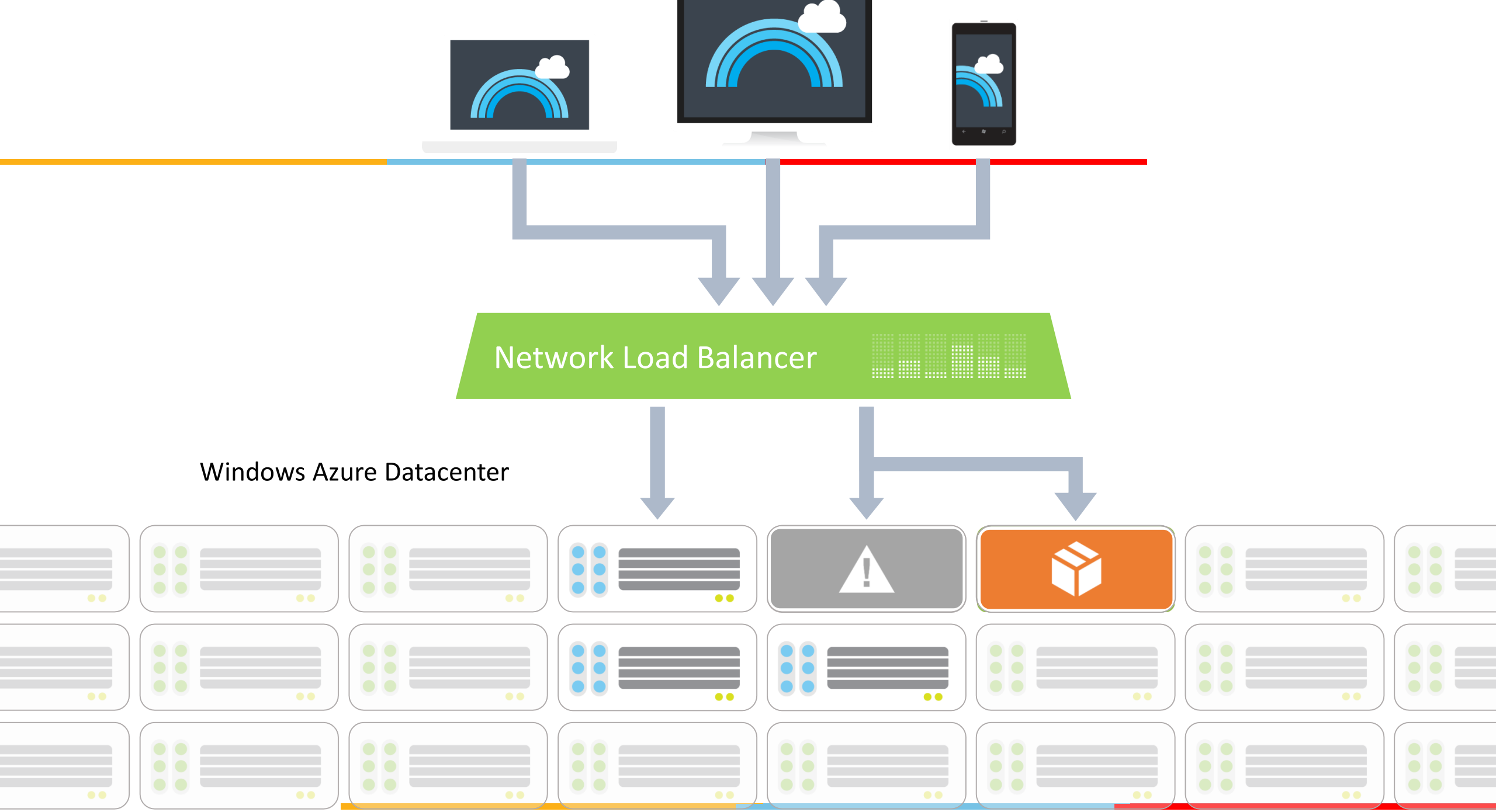
Windows Azure
Datacenter

Provision Role Instances
Deploy App Code
Configure Network

Network Load Balancer

← Network load-balancer configured for traffic

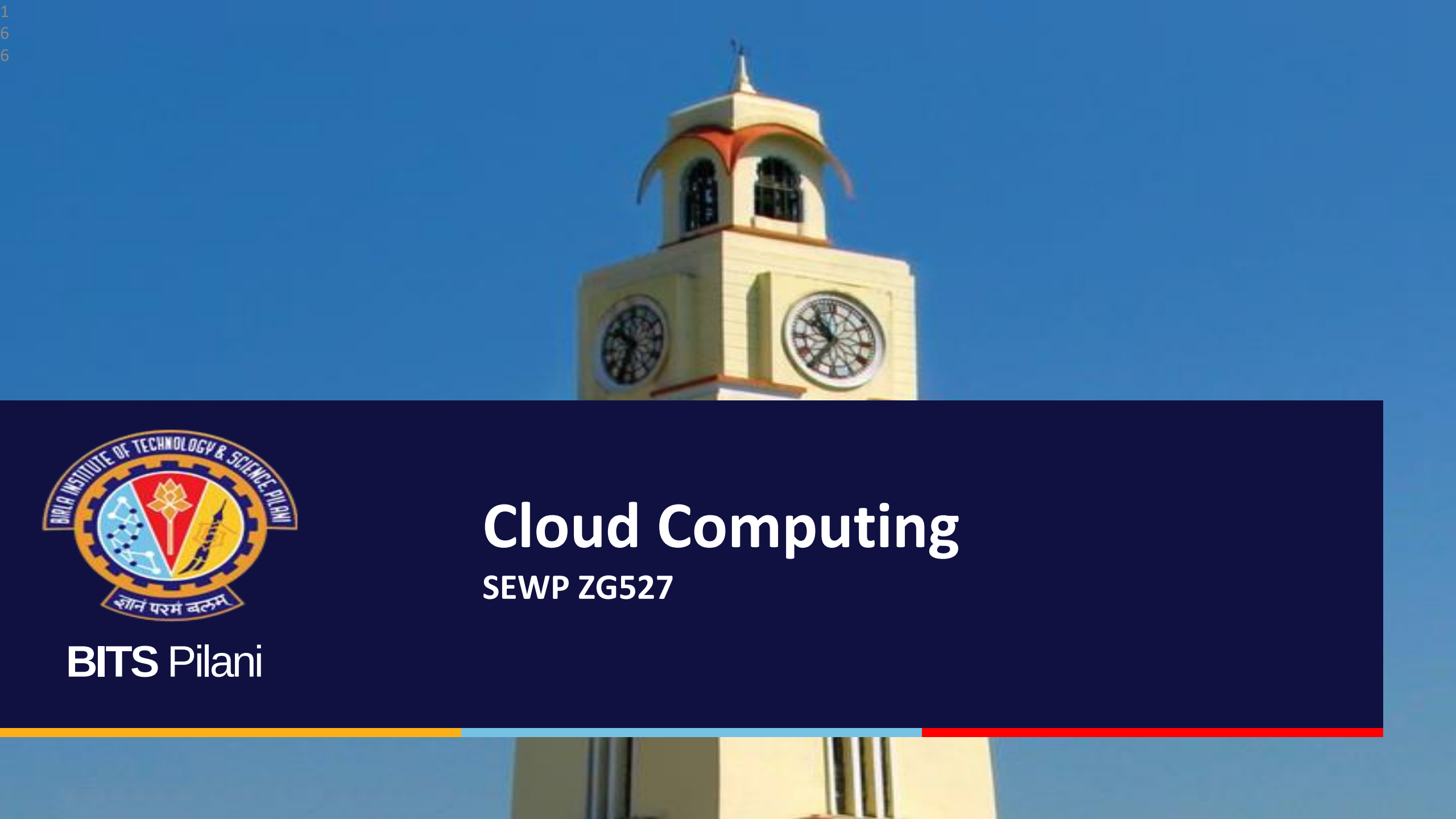
Windows Azure Datacenter


Network Load Balancer
Windows Azure Datacenter

Paas Vendors

- Common PaaS vendors include Salesforce.com's Force.com, which provides an enterprise customer relationship management (CRM) platform. PaaS platforms for software development and management include Appear IQ, Mendix, Amazon Web Services (AWS) Elastic Beanstalk, Google App Engine and Heroku.

THANK YOU

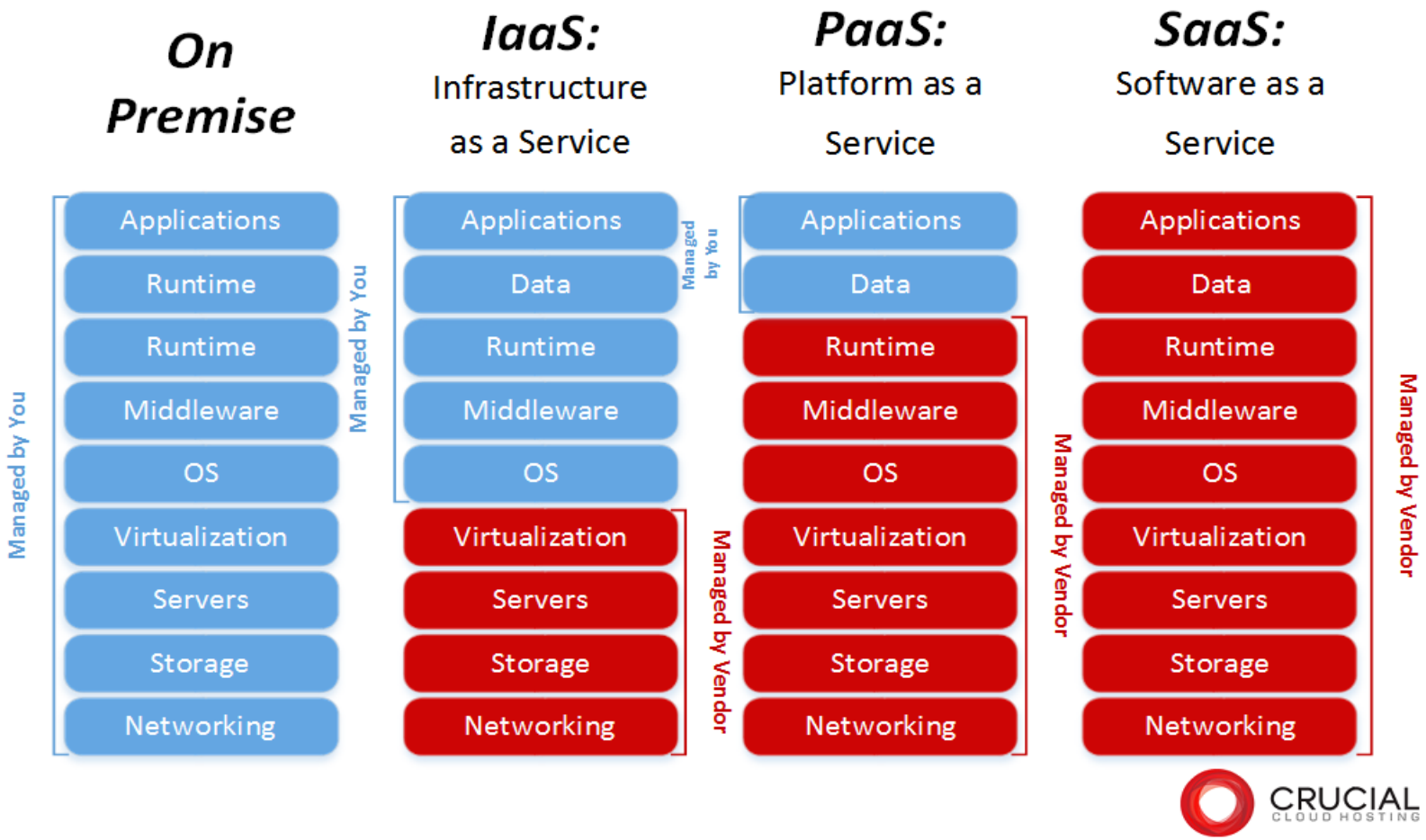


BITS Pilani

Cloud Computing

SEWP ZG527

Dependency on IaaS and PaaS



What is SaaS?

- Software as a service is a software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet.

- Shortly, in the SaaS model software is deployed as a hosted service and accessed over the Internet, as opposed to “On Premise.”

- Software delivered to home consumers, small business, medium and large business
  - The traditional model of software distribution, in which software is purchased for and installed on personal computers, is sometimes referred to as software as a product.

Problems in traditional Model

- In the traditional model of software delivery, the customer acquires a perpetual license and assumes responsibility for managing the software.
- There is a high upfront cost associated with the purchase of the license, as well as the burden of implementation and ongoing maintenance.
- ROI is often delayed considerably, and, due to the rapid pace of technological change, expensive software solutions can quickly become obsolete.

Problems in traditional Model

Traditional Software

Build Your Own

On-Demand Utility

Plug In, Subscribe

Pay-per-Use

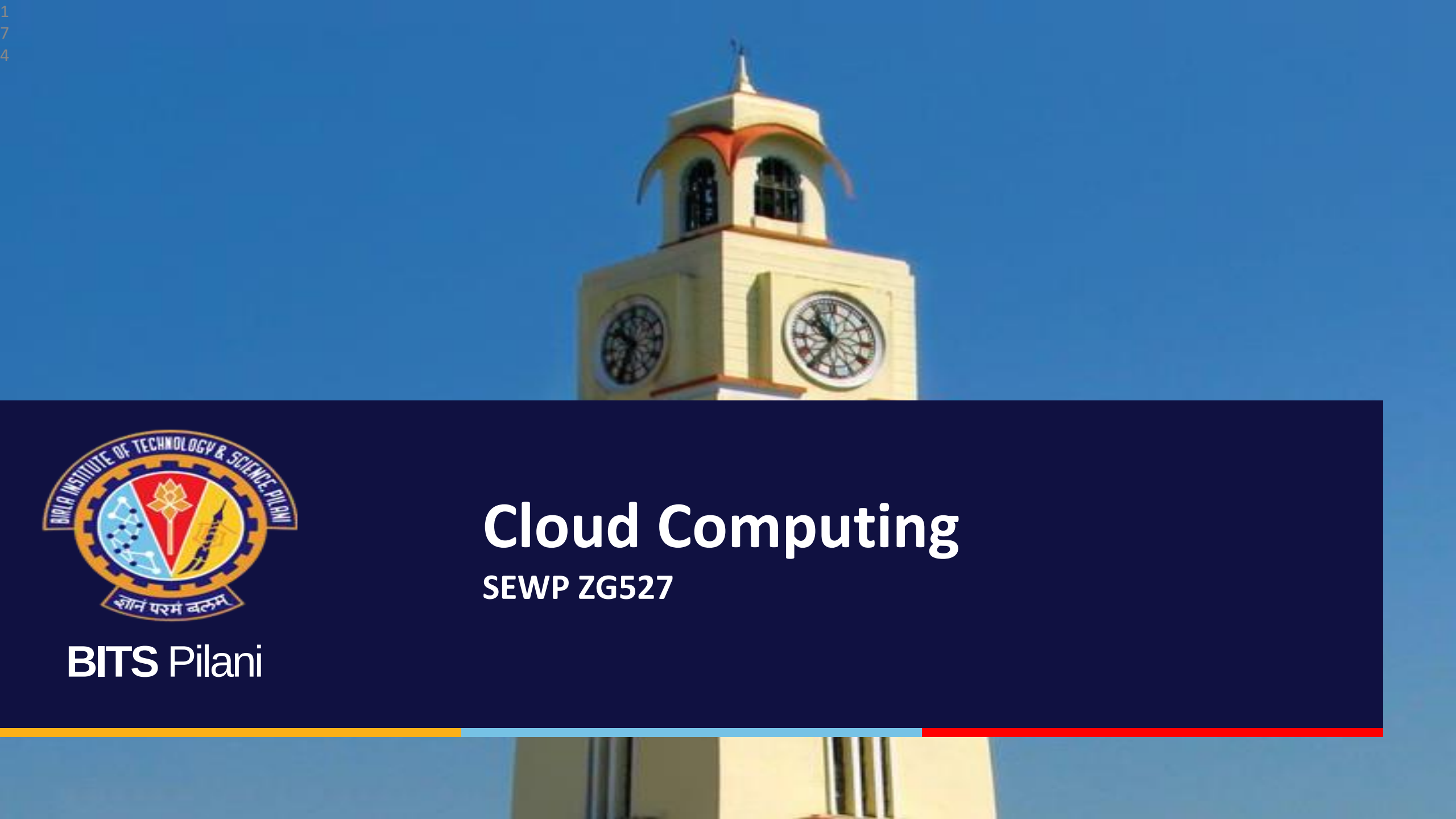
SaaS – How is it delivered

- The web as a platform is the center point. The web as a platform is the center point
- Network-based access to, and management of, commercially available (i.e., not custom) software application delivery that typically is closer to a one-to-many model (single instance, multi-tenant architecture) than to a one-to-one model, including architecture, pricing, partnering, and management characteristics

- Software delivered to home consumers, small business, medium and large business
    - The traditional model of software distribution, in which software is purchased for and installed on personal computers, is sometimes referred to as software as a product.

Thanks!!!

Queries?



BITS Pilani

Cloud Computing

SEWP ZG527

SaaS – Architecture

- Run by
  - Bandwidth technologies
  - The cost of a PC has been reduced significantly with more powerful computing but the cost of application software has not followed
  - Timely and expensive setup and maintenance costs
  - Licensing issues for business are contributing significantly to the use of illegal software and piracy.

SaaS Application Architecture

- Scalable
- Multitenant efficient
- Configurable

- **Scaling the application** - maximizing concurrency, and using application resources more efficiently
- i.e. optimizing locking duration, statelessness, sharing pooled resources such as threads and network connections, caching reference data, and partitioning large databases.

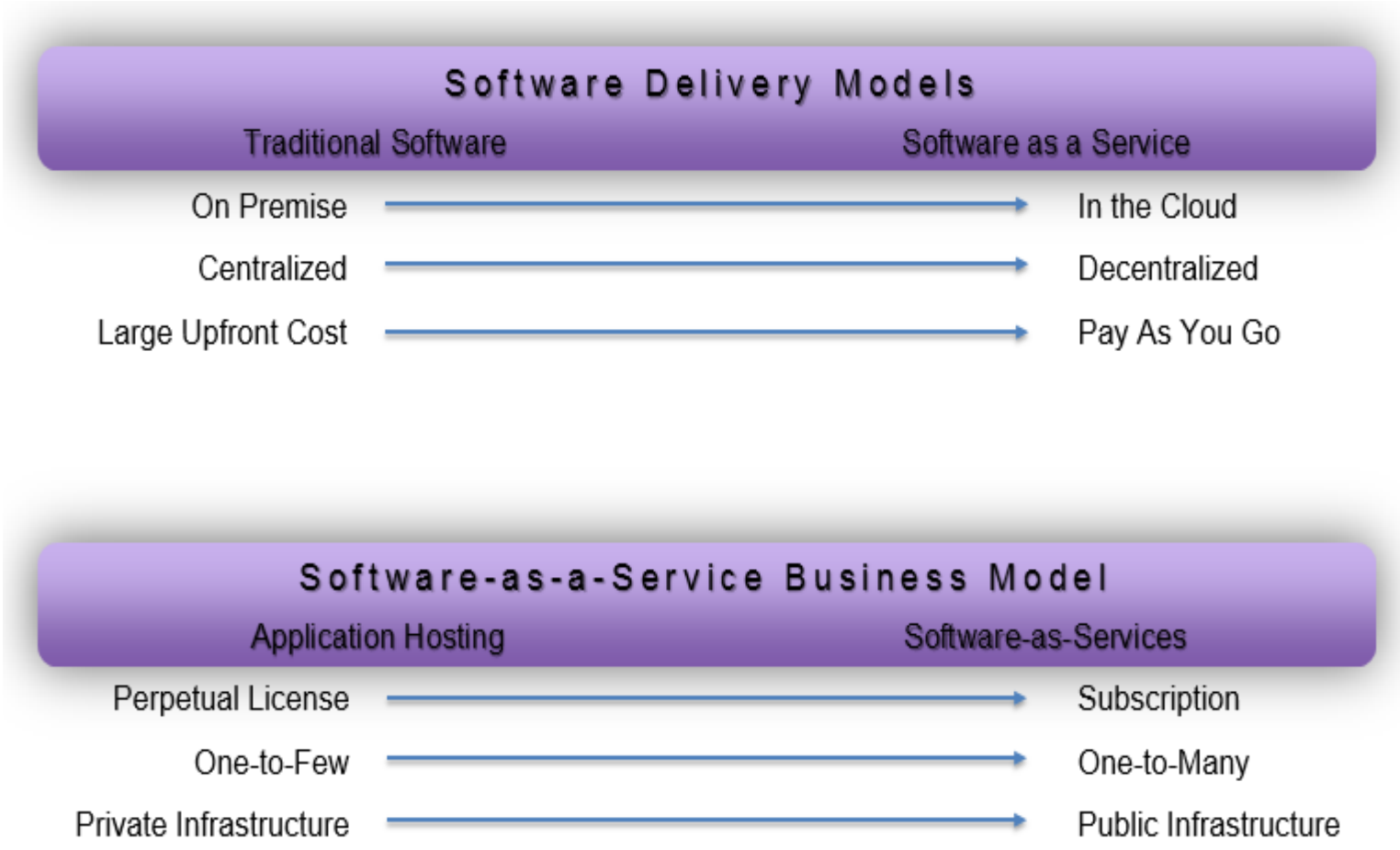
SaaS Application Architecture

- **Multi-tenancy** – important architectural shift from designing isolated, single-tenant applications
- One application instance must be able to accommodate users from multiple other companies at the same time
- All transparent to any of the users.
- This requires an architecture that maximizes the sharing of resources across tenants
- is still able to differentiate data belonging to different customers.

SaaS Application Architecture

- **Configurable** - a single application instance on a single server has to accommodate users from several different companies at once
- To customize the application for one customer will change the application for other customers as well.
- Traditionally customizing an application would mean code changes
- Each customer uses metadata to configure the way the application appears and behaves for its users.
- Customers configuring applications must be simple and easy without incurring extra development or operation costs

SaaS Models


Software Delivery Models
Traditional Software
Software as a Service
On Premise
In the Cloud
Centralized
Decentralized
Large Upfront Cost
Pay As You Go
Software-as-a-Service Business Model
Application Hosting
Software-as-Services
Perpetual License
Subscription
One-to-Few
One-to-Many
Private Infrastructure
Public Infrastructure

Business Model comparisons

Traditional packaged software	Software as a service
Designed for customers to install, manage and maintain.	Designed from the outset up for delivery as Internet-based services
Architect solutions to be run by an individual company in a dedicated instantiation of the software	Designed to run thousands of different customers on a single code

Business Model comparisons

Traditional packaged Software

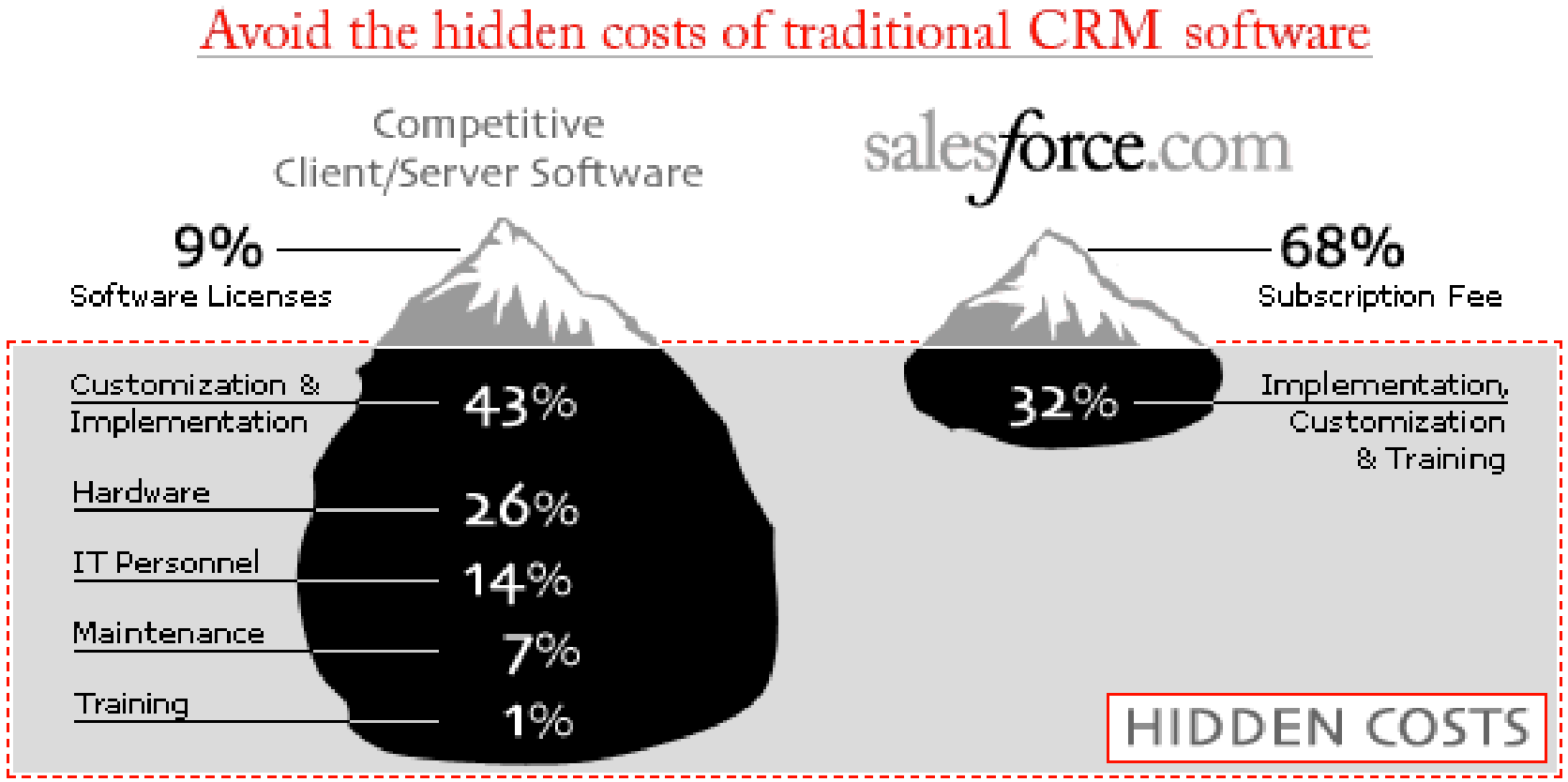
- Infrequent, major upgrades every 18-24 months, sold individually to each installed base customer.
- Version control
- Upgrade fee
- Streamlined, repeatable functionality via Web services, open APIs and standard connectors

Software as a serivce

- Frequent, "digestible" upgrades every 3-6 months to minimize customer disruption and enhance satisfaction.
- Fixing a problem for one customer fixes it for everyone
- May use open APIs and Web services to facilitate integration, but each customer must typically pay for one-off integration work.

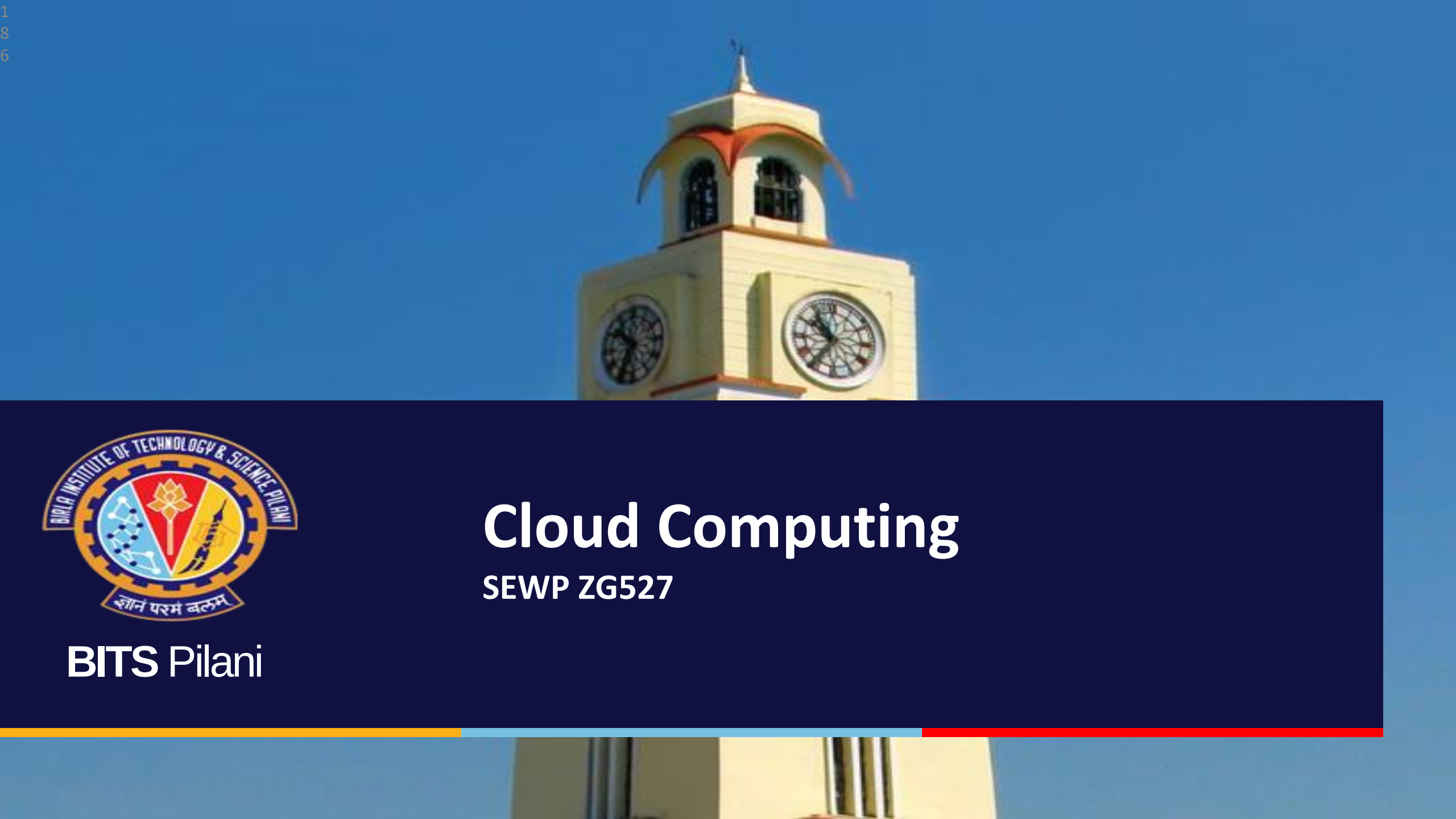
Business Model comparisons

Hidden Cost



Thanks!!!

Queries?



Cloud Computing

SEWP ZG527

BITS Pilani

SaaS Advantages

Characteristics	Benefits
Network delivered access to commercially available software	**No local infrastructure or software to purchase or maintain** Applications & data are available anywhere with network connectivity
Application delivery is one-to-many model	Operating costs are reduced by managing infrastructure in central locations rather than at each customer's site
Built on optimized & robust platform	Improved availability and reliability
Customer pays for as much as they need when they need it	Lower TCO

Attributes	Software-as-a-Service (SaaS)	On-Premise
Alternate labels	On-Demand, Subscription Based, Hosted, Application Service Provider (ASP)	Installed, Hosted On-Premise
Purchase model	Lease, rent, subscription	Most commonly purchasing of licenses (ownership) with some lease and rent options
Maintenance and support	Typically included	For purchase option it would be based on the percentage of license fees
Security	Typically in a 3rd party highly secure data center	Responsibility of the costumer
Upgrades	Typically included	Included with maintenance
Data model	Shared (multi-tenant) or dedicated (single) instance depending on the vendor	Dedicated (single) instance on the customer's servers
Access	Typically all major web browsers	Typically all major web browsers
Initial investment	Low upfront cost since no hardware or infrastructure investment	Higher cost since typically purchasing licenses and hardware
3-5 year investment	Reoccurring fee	One-time fee
Legal implications	Hosted by 3rd party and lack of ownership. This needs to be clearly defined in a SLA	Less of an issue with the purchase option where costumer claims ownership
Implementation	Typically shorter since all is hosted by the vendor	Can be lengthier since it is installed and integrated with the existing infrastructure
Integration	Can be limited due to Web standards and protocols	Tends to be more flexible since it resides behind your firewall on your servers

Applicability – Scenario 1

Single-User software application

- Organize personal information
- Run on users' own local computer
- Serve only one user at a time
- Inapplicable to Saas model
  - Data security issue
  - Network performance issue
- Example: Microsoft office suite

Applicability – Scenario 2

Infrastructure software

- Serve as the foundation for most other enterprise software application
- Inapplicable to Saas model
  - Installation locally is required
  - Form the basis to run other application
- Example: Window XP, Oracle database

Applicability – Scenario 3

Embedded Software

- Software component for embedded system
- Support the functionality of the hardware device
- Inapplicable to Saas model
  - Embedded software and hardware is combined together and is inseparable
- Example: software embedded in ATM machines, cell phones, routers, medical equipment, etc

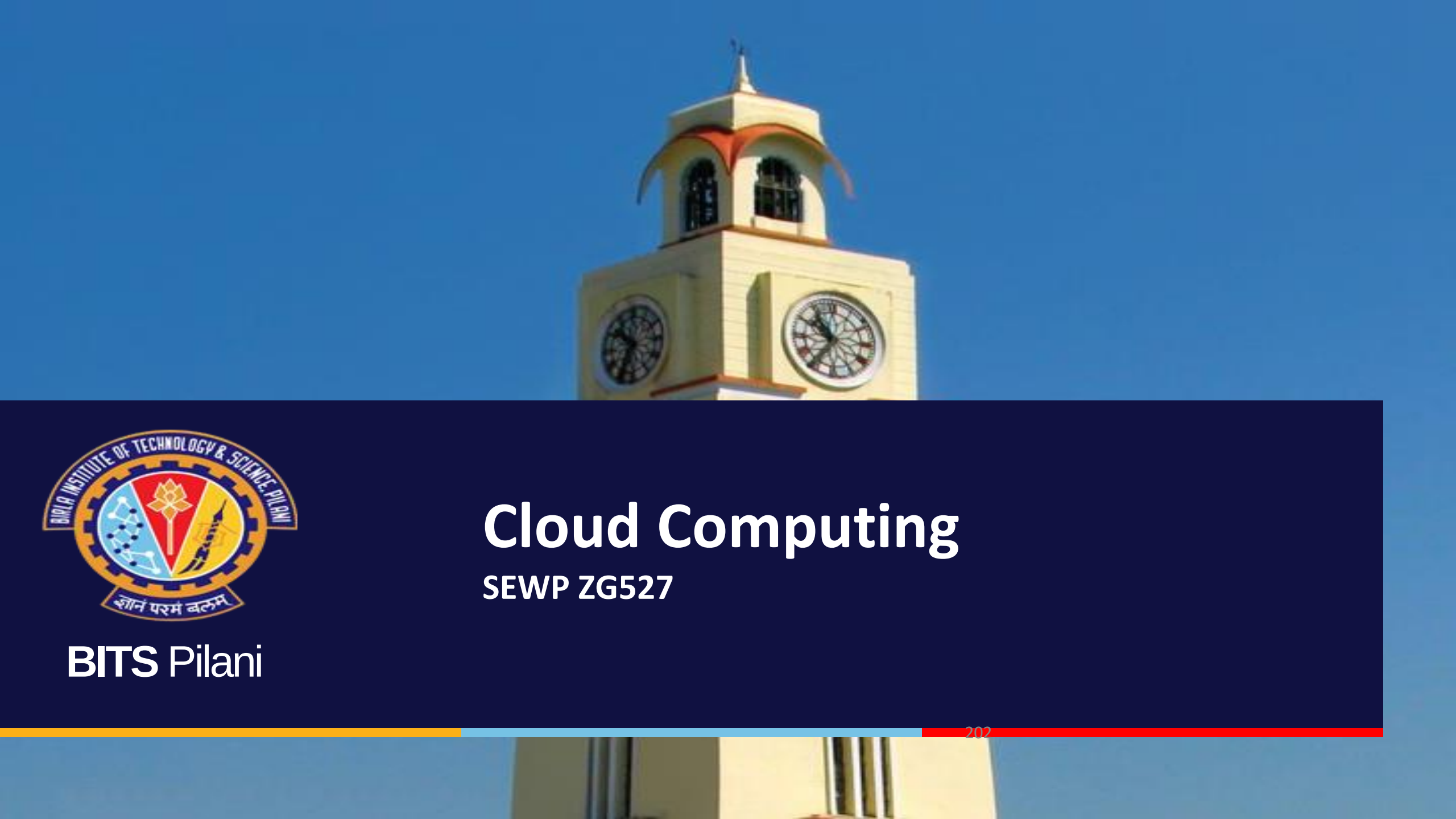
Applicability – Scenario 4

Enterprise Software Application

- Perform business functions
- Organize internal and external information
- Share data among internal and external users
- The most standard type of software applicable to Saas model
- Example: Saleforce.com CRM application, Siebel On-demand application

Thanks!!!

Queries?

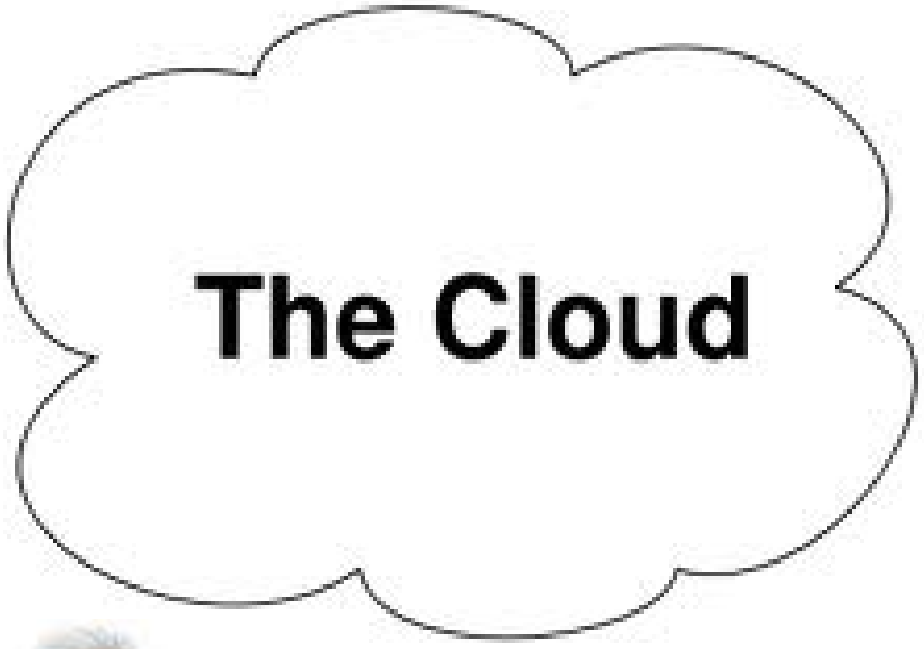


BITS Pilani

Cloud Computing

SEWP ZG527

Cloud ? Th  ooks easy!!!

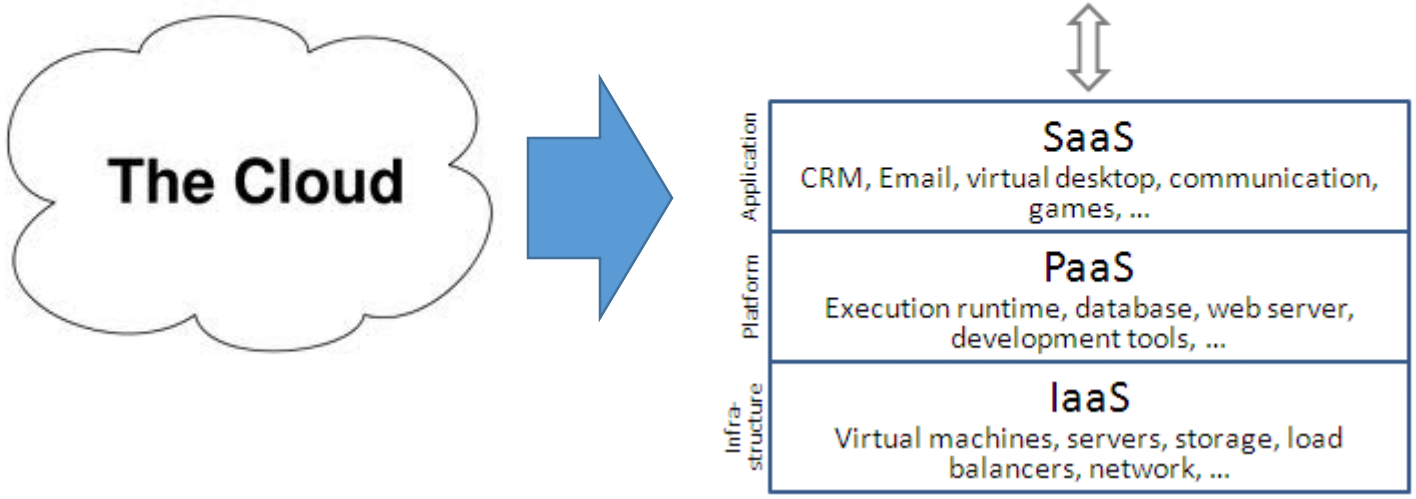


Is it So?????

Introduction

So CLOUD requires managing……

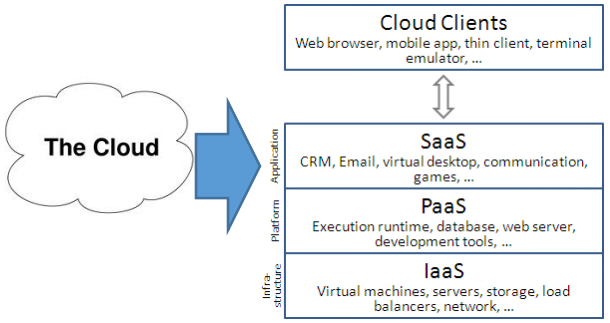
But what will you manage

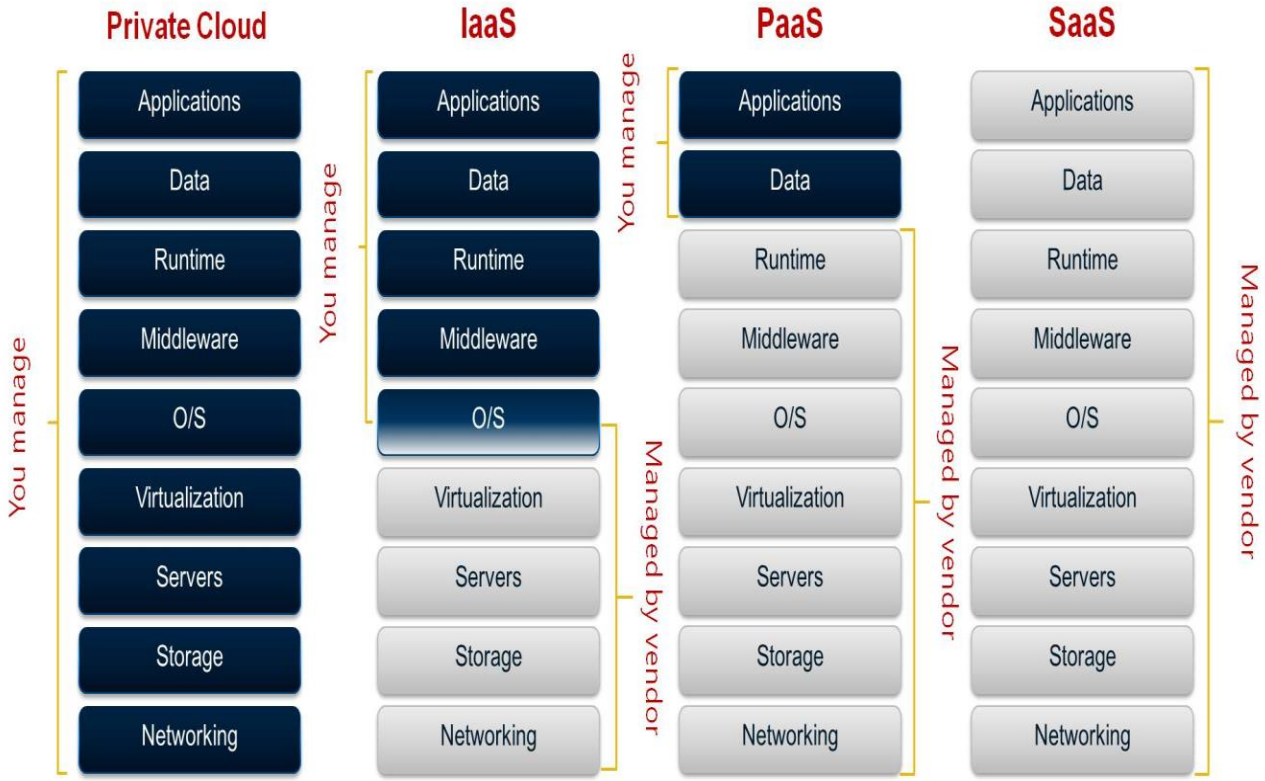


Introduction

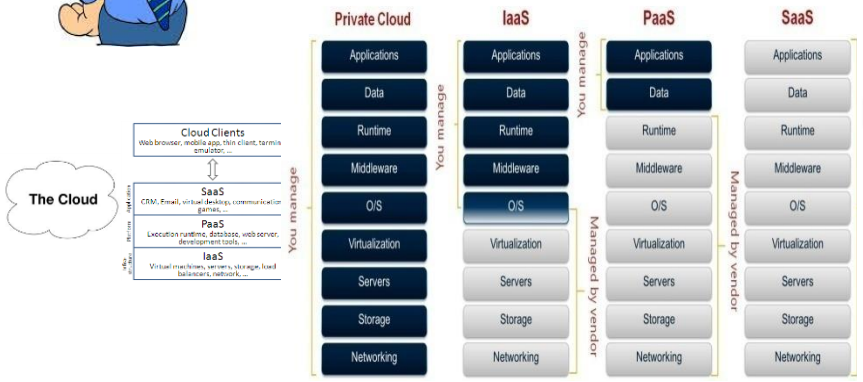
Ok, ok got to know what to manage…..

Is it so……..????

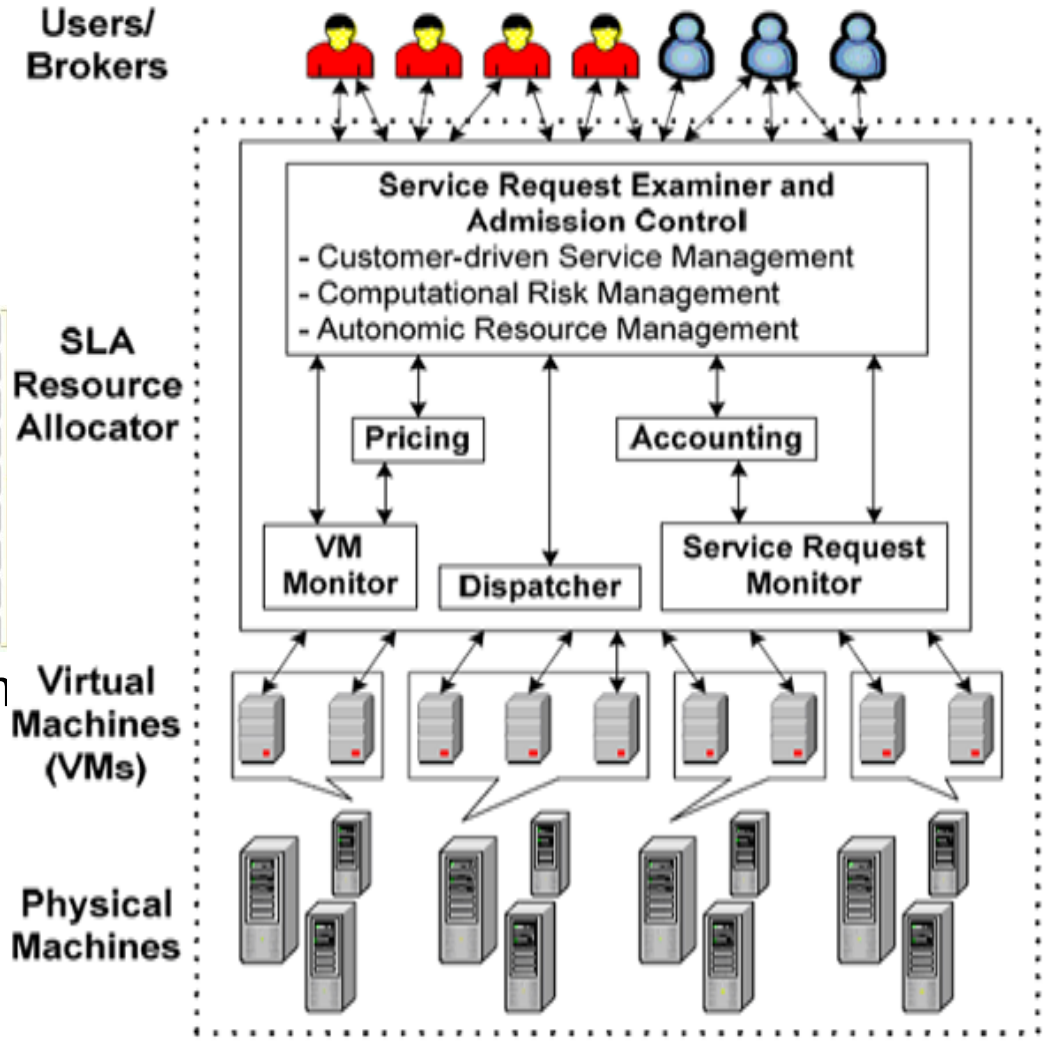




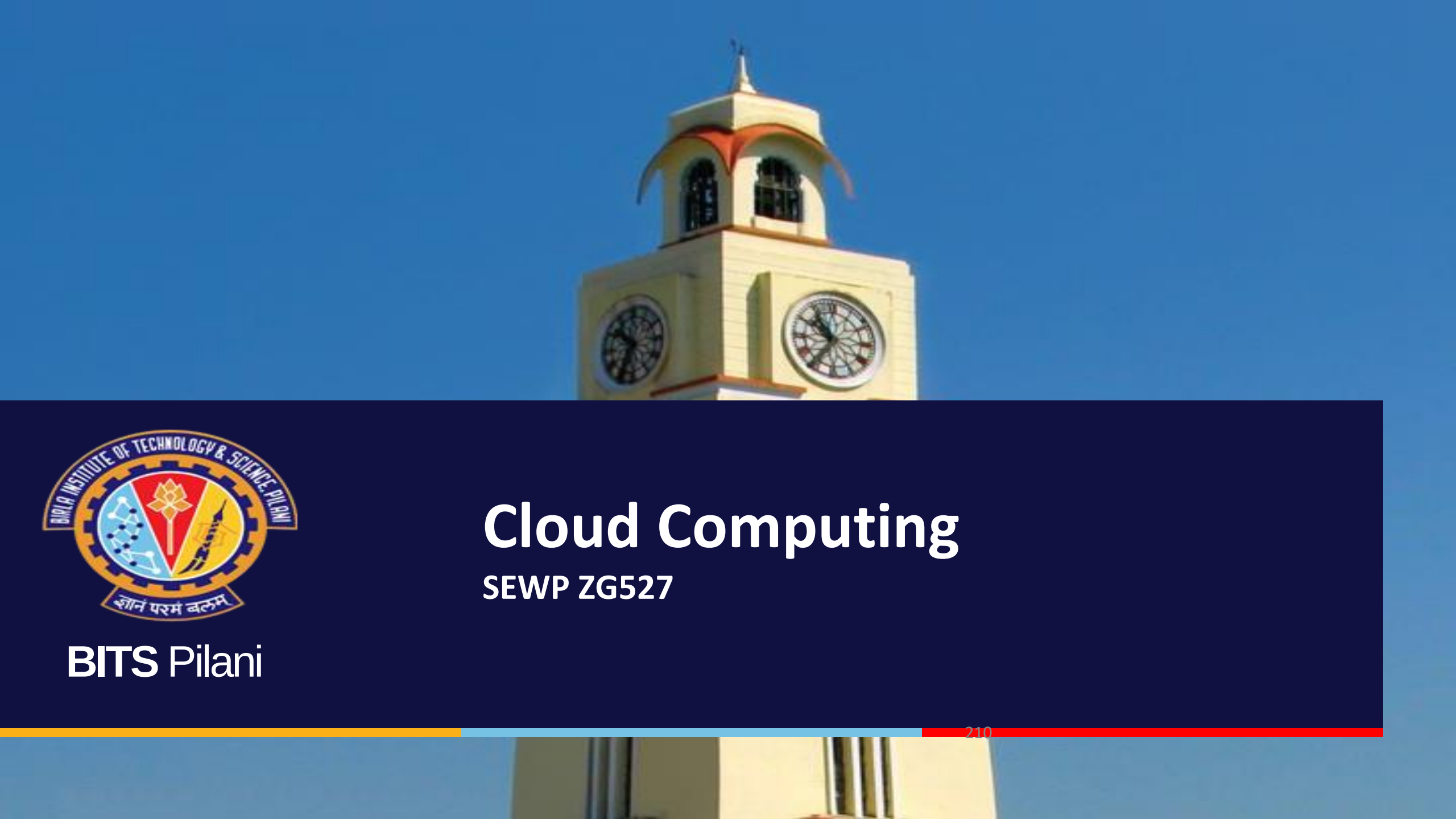
OMG!!!!



Therefore the key requirem for cloud architecture is "**efficient management**" of resources at all the three layers of cloud stack



- Cloud **Distributed environment**
  - With large scale of systems to manage
  - Support of multi-tenancy
  - Management to maintain SLAs
- So there is need for **automation** to replace manual operations and to reduce overall cost



BITS Pilani

Cloud Computing

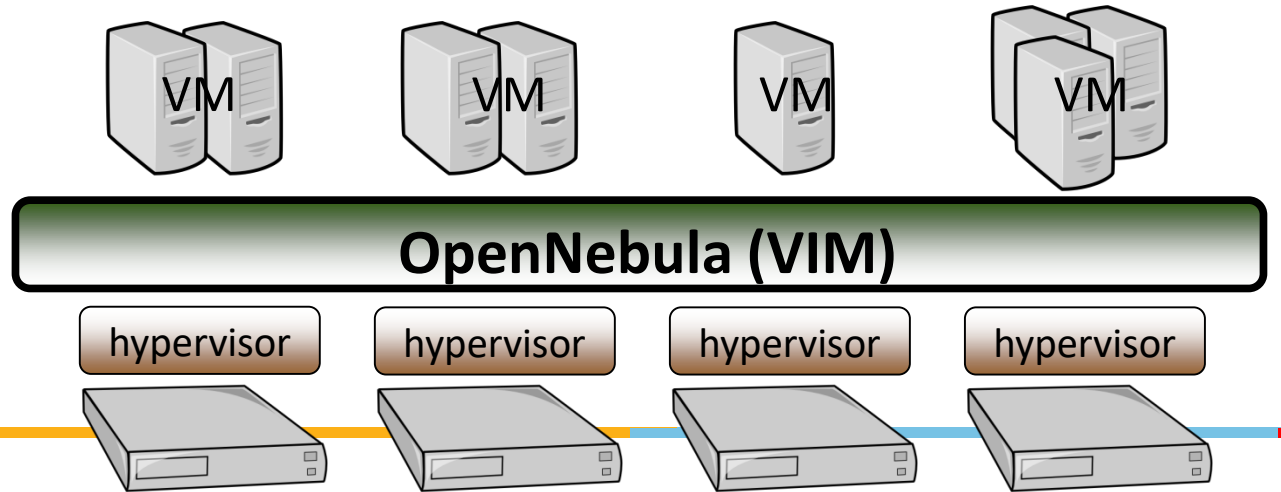
SEWP ZG527

Need of the hour???

Virtual Infrastructure Managers

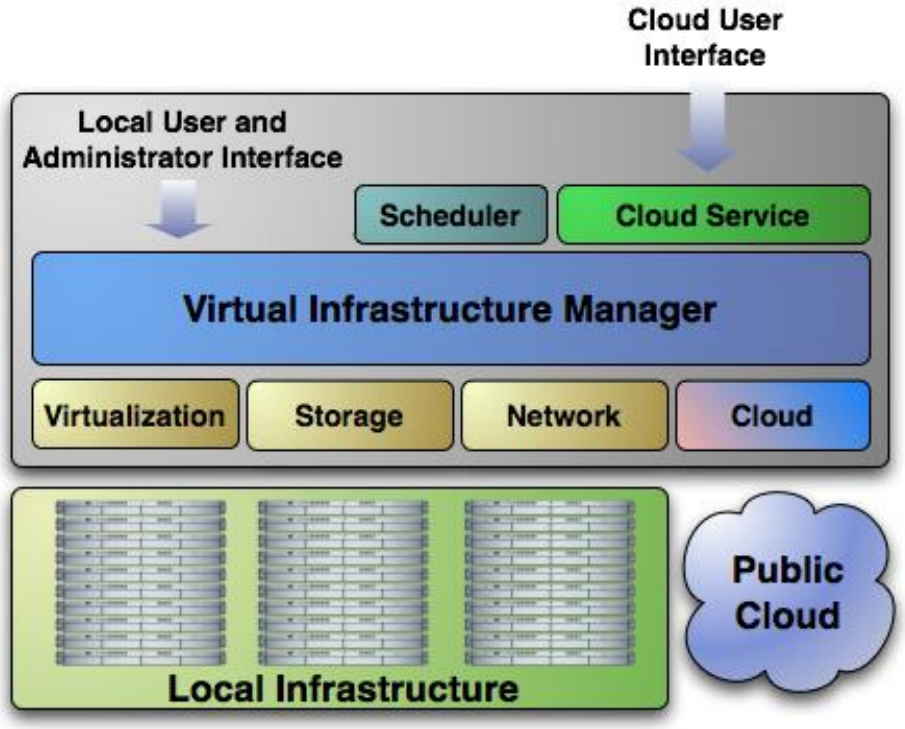
Why a Virtual Infrastructure Manager?

- VMs are great!!...but something more is needed
  - Where did/do I put my VM? (***scheduling & monitoring***
  - How do I provision a new cluster node? (***clone***)
  - What IP addresses are available? (***networking)***
- Provide a ***uniform view*** of the resource pool
- ***Life-cycle management*** and monitoring of VM
- The VIM should ***integrate*** Image, Network and Virtualization



Extending the Benefits of Virtualization to Clusters

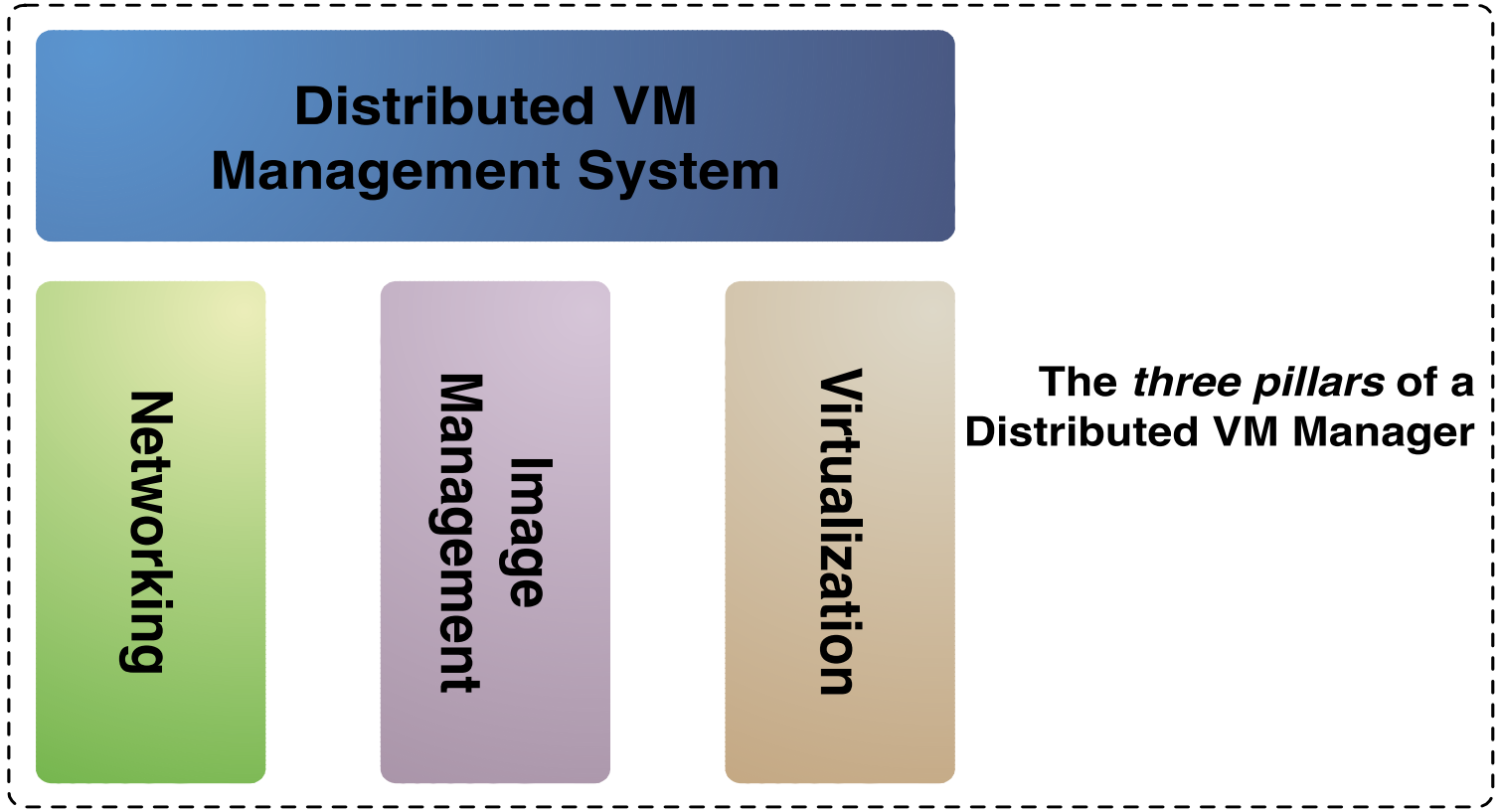
- Dynamic deployment and re-placement of virtual machines on a pool of physical resources
- Transform a rigid distributed physical infrastructure into a flexible and agile virtual infrastructure

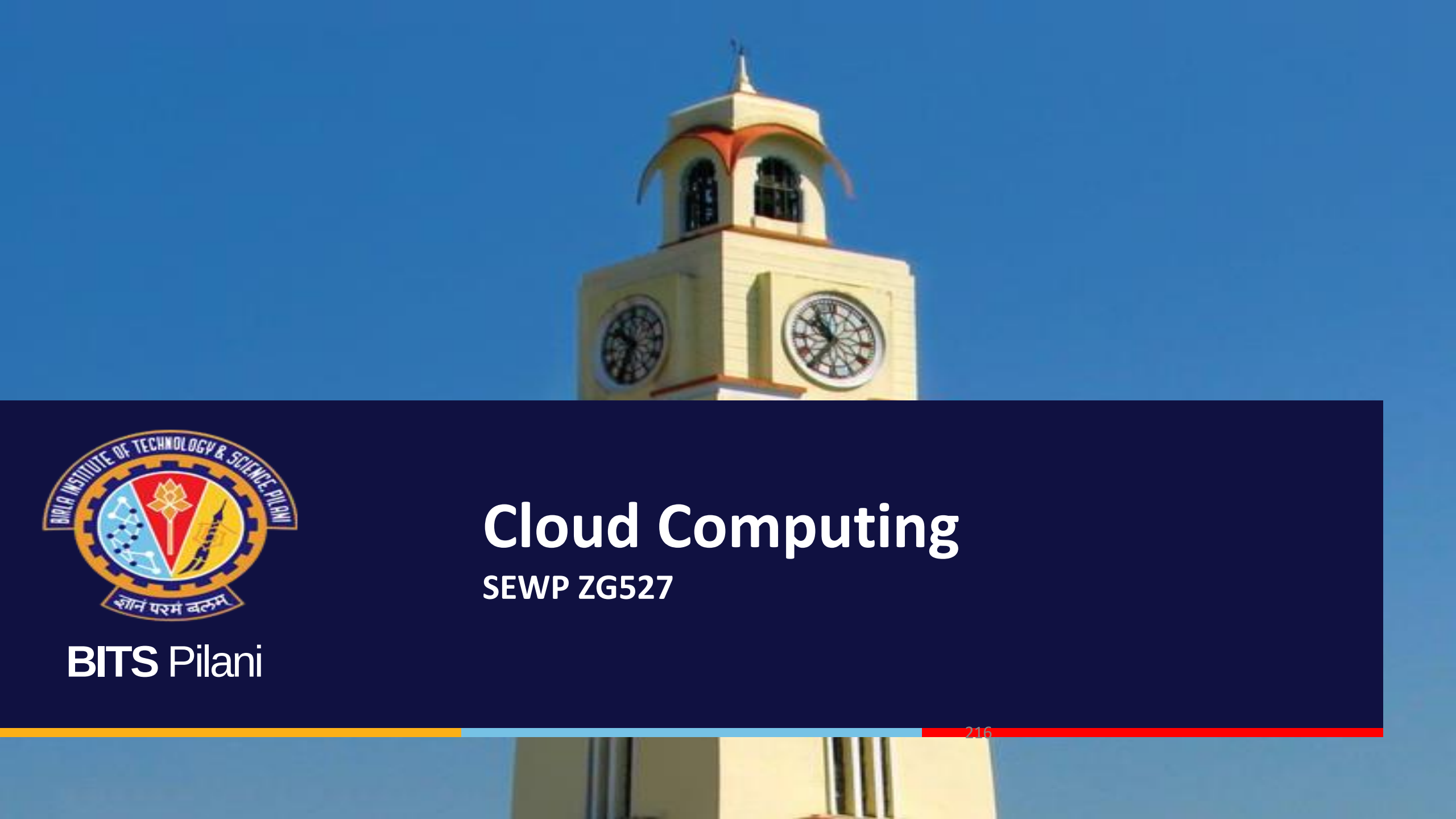


- Backend of Public Cloud: Internal management of the infrastructure
- Private Cloud: Virtualization of cluster or data-center for internal users
- Cloud Interoperation: On-demand access to public clouds

Virtual Machine Management Model

Distributed VM Management Model





BITS Pilani

Cloud Computing

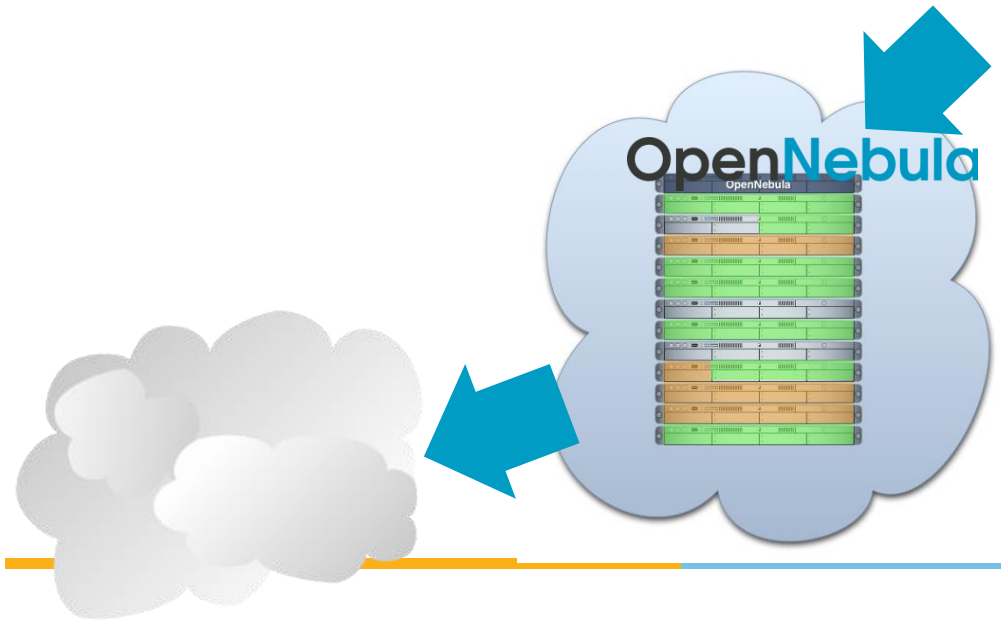
SEWP ZG527

What is OpenNebula?

What is OpenNebula?

Enabling Technology to Build your Cloud

- Private Cloud to simplify and optimize internal operations
- Hybrid Cloud to supplement the capacity of the Private Cloud
- Public Cloud to expose your Private to external users



The Benefits of OpenNebula

For the Infrastructure Manager

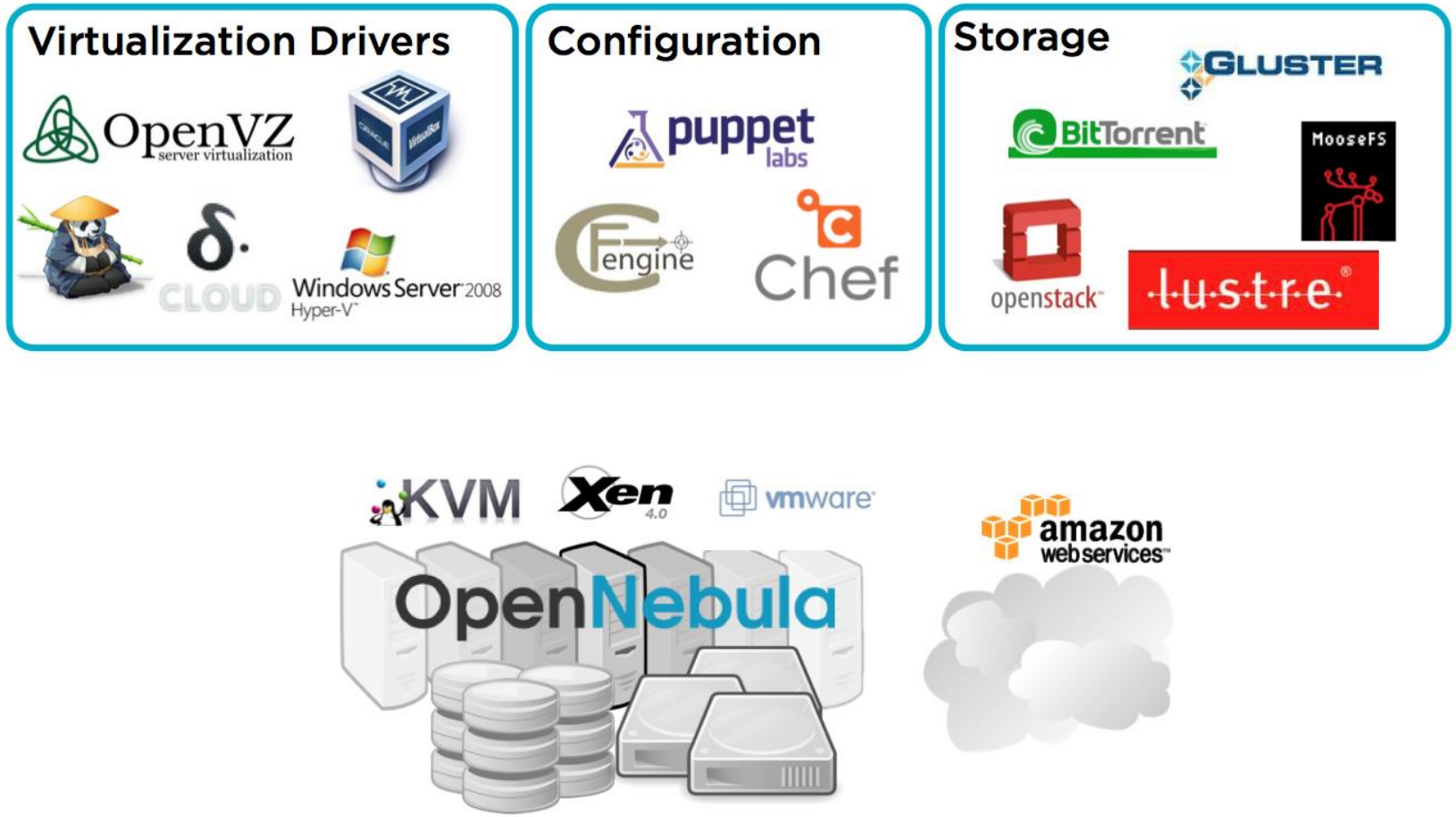
- Centralized management of VM workload and distributed infrastructures
- Support for VM placement policies: balance of workload, server consolidation…
- Dynamic resizing of the infrastructure
- Dynamic partition and isolation of clusters
- Dynamic scaling of private infrastructure to meet fluctuating demands
- Lower infrastructure expenses combining local and remote Cloud resources

For the Infrastructure User

- Faster delivery and scalability of services
- Support for heterogeneous execution environments
- Full control of the lifecycle of virtualized services management

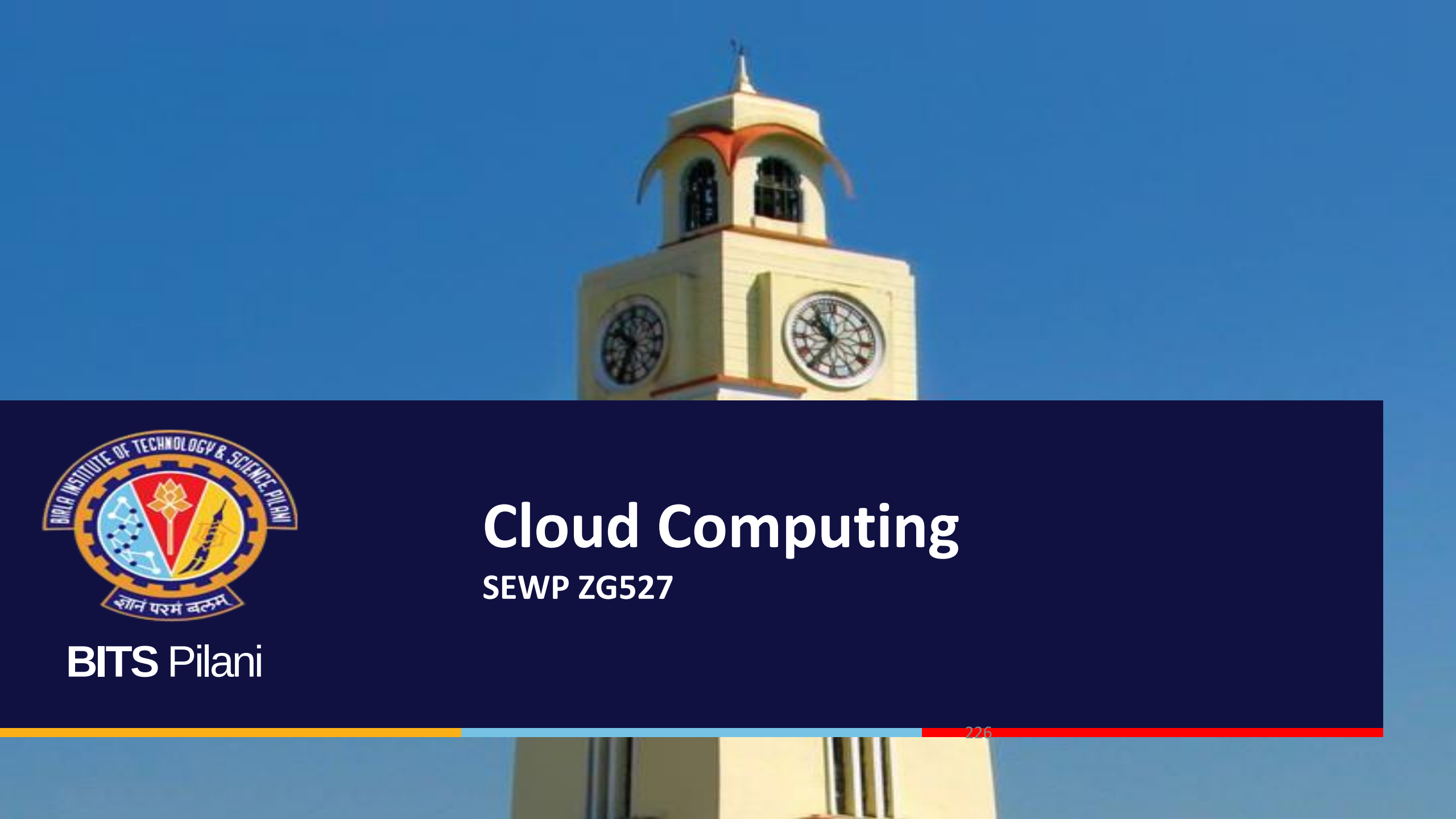
Interoperability from the Cloud Provider perspective

Interoperable (platform independent), innovative (feature-rich) and proven (mature to run in production).



The main features of OpenNebula

Feature	Function
Internal Interface	• Unix-like CLI for fully management of VM life-cycle and physical boxes • XML-RPC API and libvirt virtualization API
Scheduler	• Requirement/rank matchmaker allowing the definition of workload and resource-aware allocation policies • Support for advance reservation of capacity through Haizea
Virtualization Management	• Xen, KVM, and VMware • Generic libvirt connector (VirtualBox planned for 1.4.2)
Image Management	• General mechanisms to transfer and clone VM images
Network Management	• Definition of isolated virtual networks to interconnect VMs
Service Management and Contextualization	• Support for multi-tier services consisting of groups of inter-connected VMs, and their auto-configuration at boot time
Security	• Management of users by the infrastructure administrator
Fault Tolerance	• Persistent database backend to store host and VM information
Scalability	• Tested in the management of medium scale infrastructures with hundreds of servers and VMs (no scalability issues has been reported)
Flexibility and Extensibility	• Open, flexible and extensible architecture, interfaces and components, allowing its integration with any product or tool



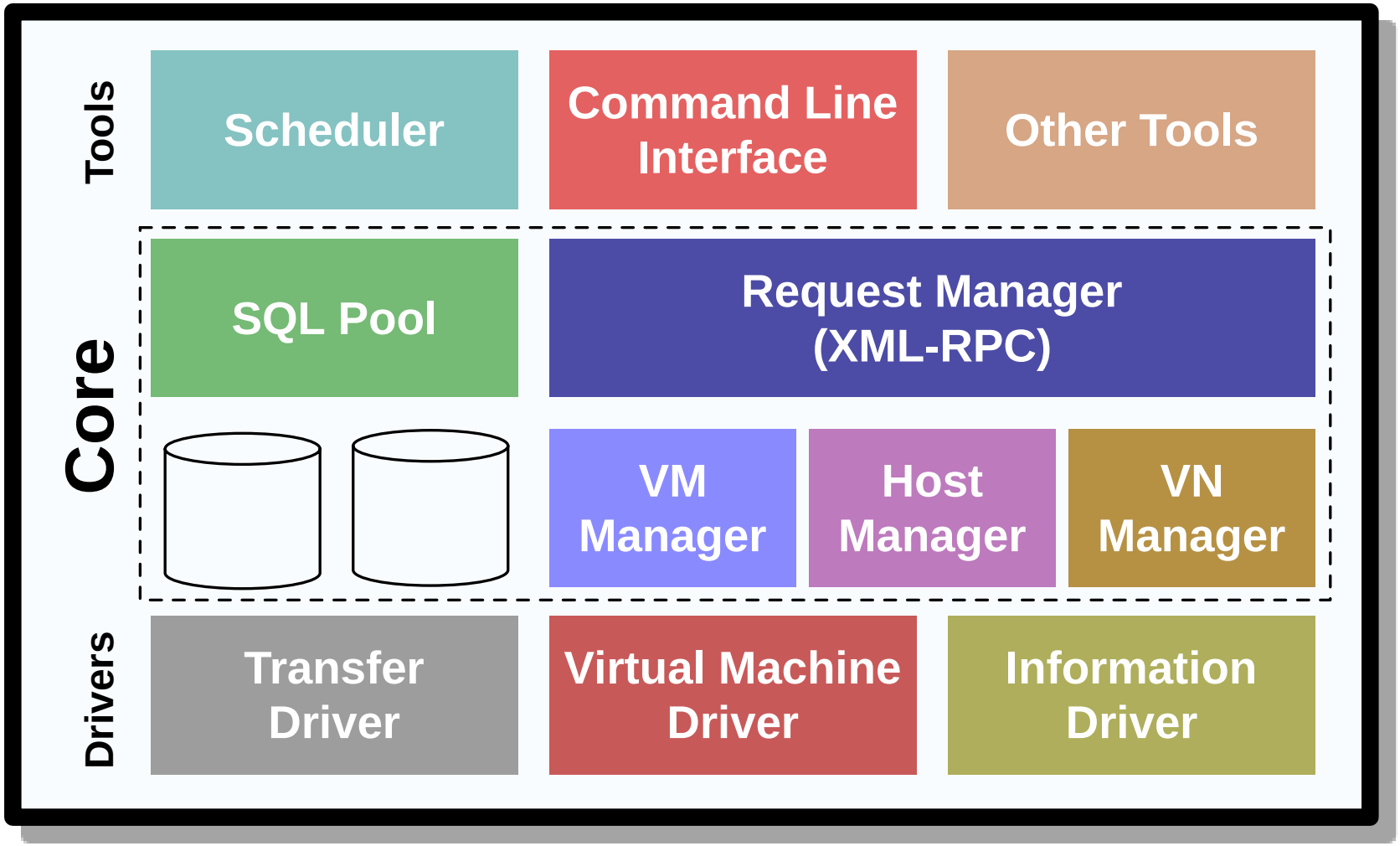
BITS Pilani

Cloud Computing

SEWP ZG527

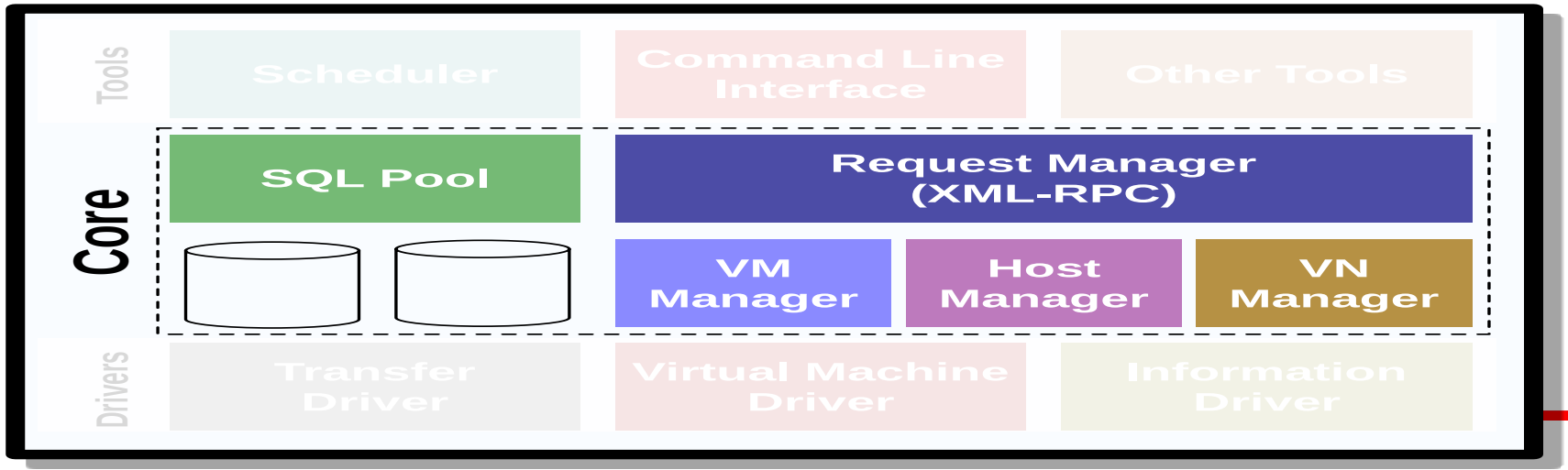
Inside OpenNebula

OpenNebula Architecture



The Core

- Request manager: Provides a XML-RPC interface to manage and get information about ONE entities.
- SQL Pool: Database that holds the state of ONE entities.
- VM Manager (virtual machine): Takes care of the VM life cycle.
- Host Manager: Holds handling information about hosts.
- VN Manager (virtual network): This component is in charge of generating MAC and IP addresses.



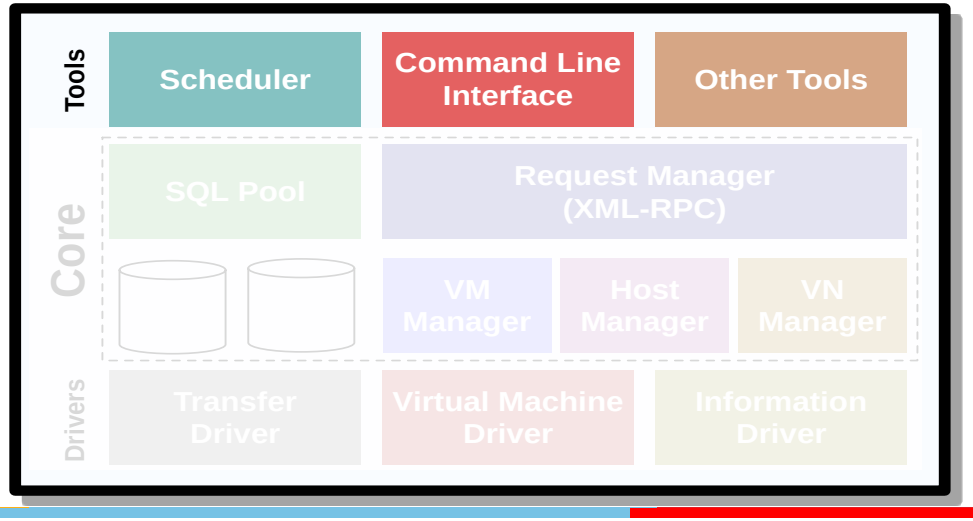
The tools layer

Scheduler:

- Searches for physical hosts to deploy newly defined VMs

Command Line Interface:

- Commands to manage OpenNebula.
- onevm: Virtual Machines
  - create, list, migrate…
- onehost: Hosts
  - create, list, disable…
- onevnet: Virtual Networks
  - create, list, delete…



The drivers layer

Transfer Driver: Takes care of the images.

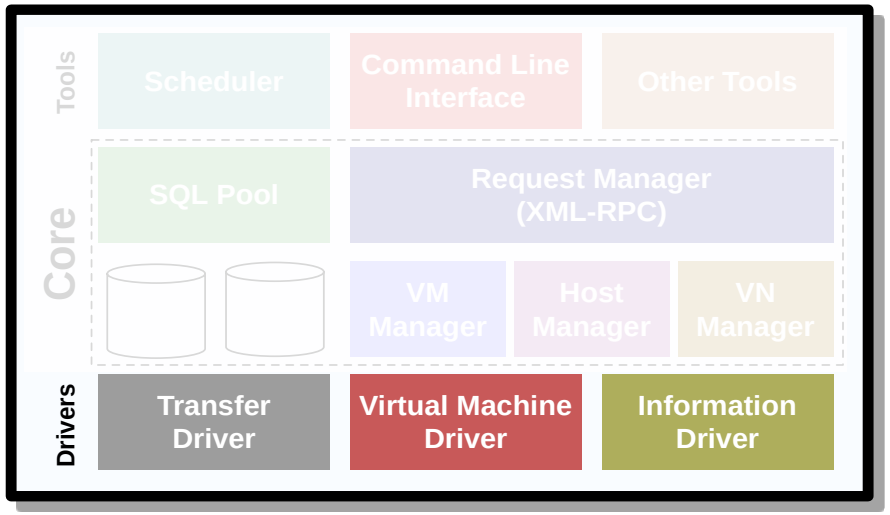
- cloning, deleting, creating swap image…

Virtual Machine Driver: Manager of the lifecycle of a virtual machine

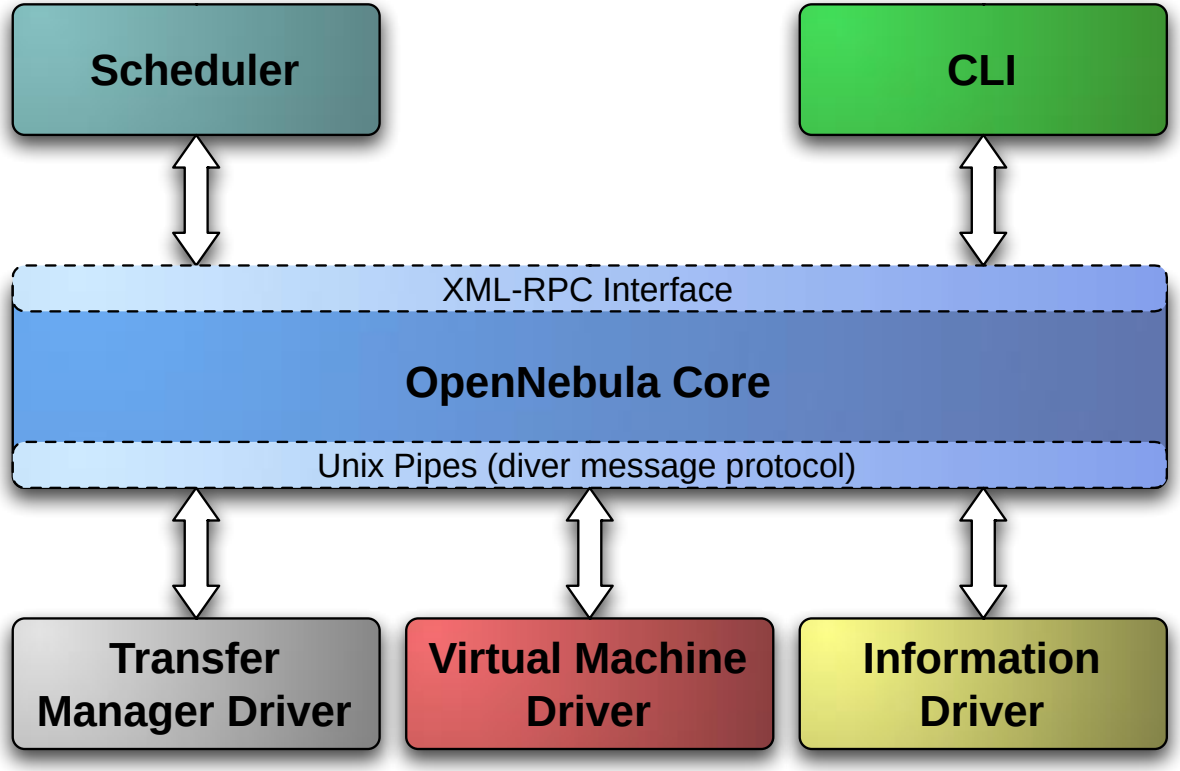
- deploy, shutdown, poll, migrate…

Information Driver: Executes scripts in physical hosts to gather information about them

- total memory, free memory, total #cpus, cpu consumed…



Process separation

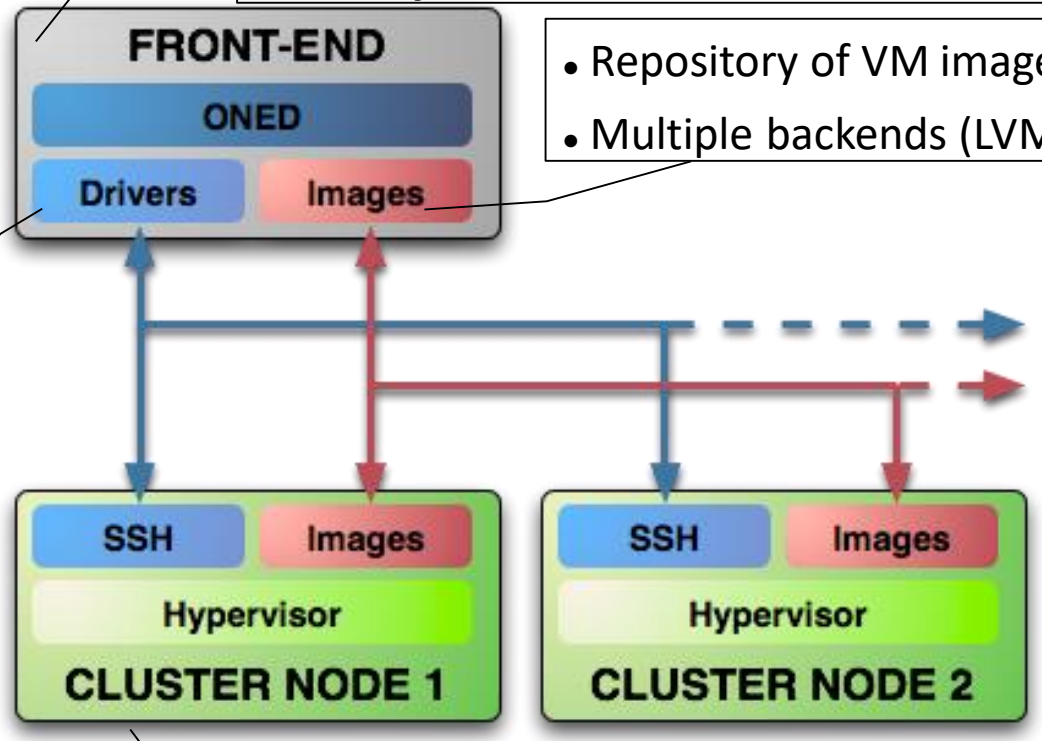


- Scheduler is a separated process, just like command line interface.
- Drivers are also separated processes using a simple text messaging protocol to communicate with OpenNebula Core Daemon (oned)

Constructing a private cloud

System Overview

- Executes the OpenNebula Services
- *Usually* acts as a classical cluster front-end



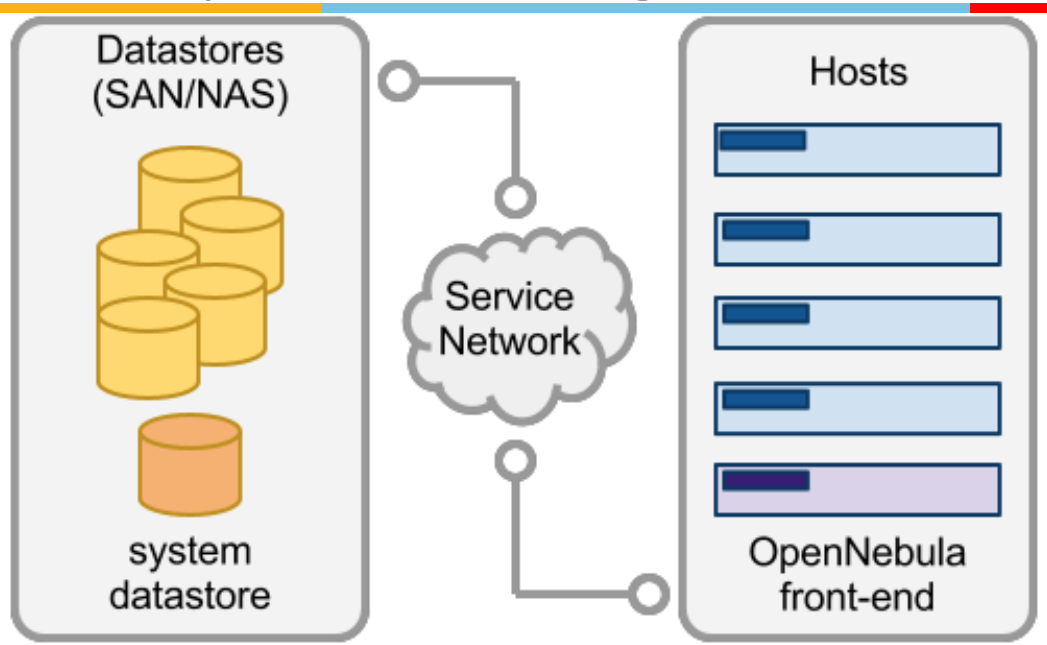
- Repository of VM images
- Multiple backends (LVM, iSCSI..)

- Modular components to interact with the cluster services
- *Types:* storage, monitoring, virtualization and network

- Provides physical resources to VMs
- *Must have* a hypervisor installed

The same host can be can be a the front-end and a node

Complex Storage behind OpenNebula



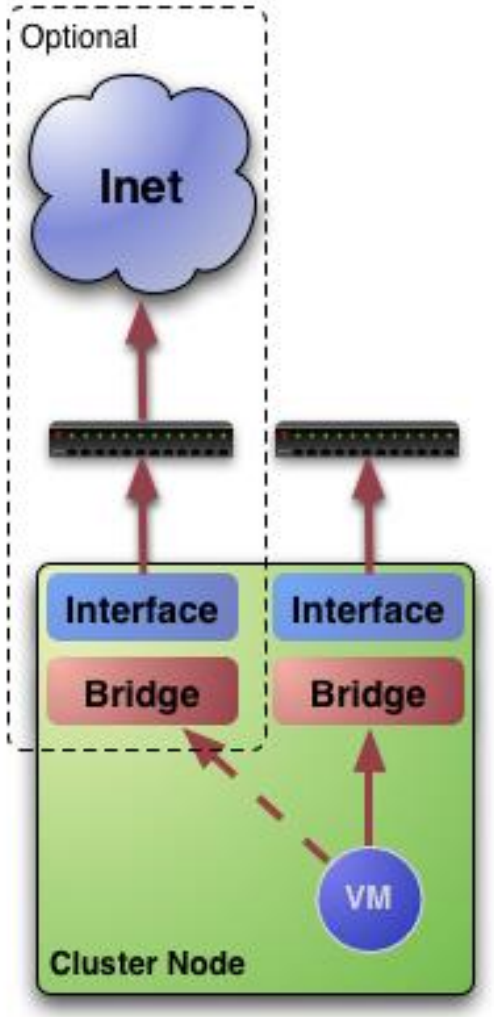
Datastore	Transfer Manager Drivers				
	shared	ssh	iscsi	qcow	vmware
System	OK	OK			
File-System	OK	OK		OK	
iSCSI			OK		
VMware	OK	OK			OK

Virtual machines and their images are represented as files

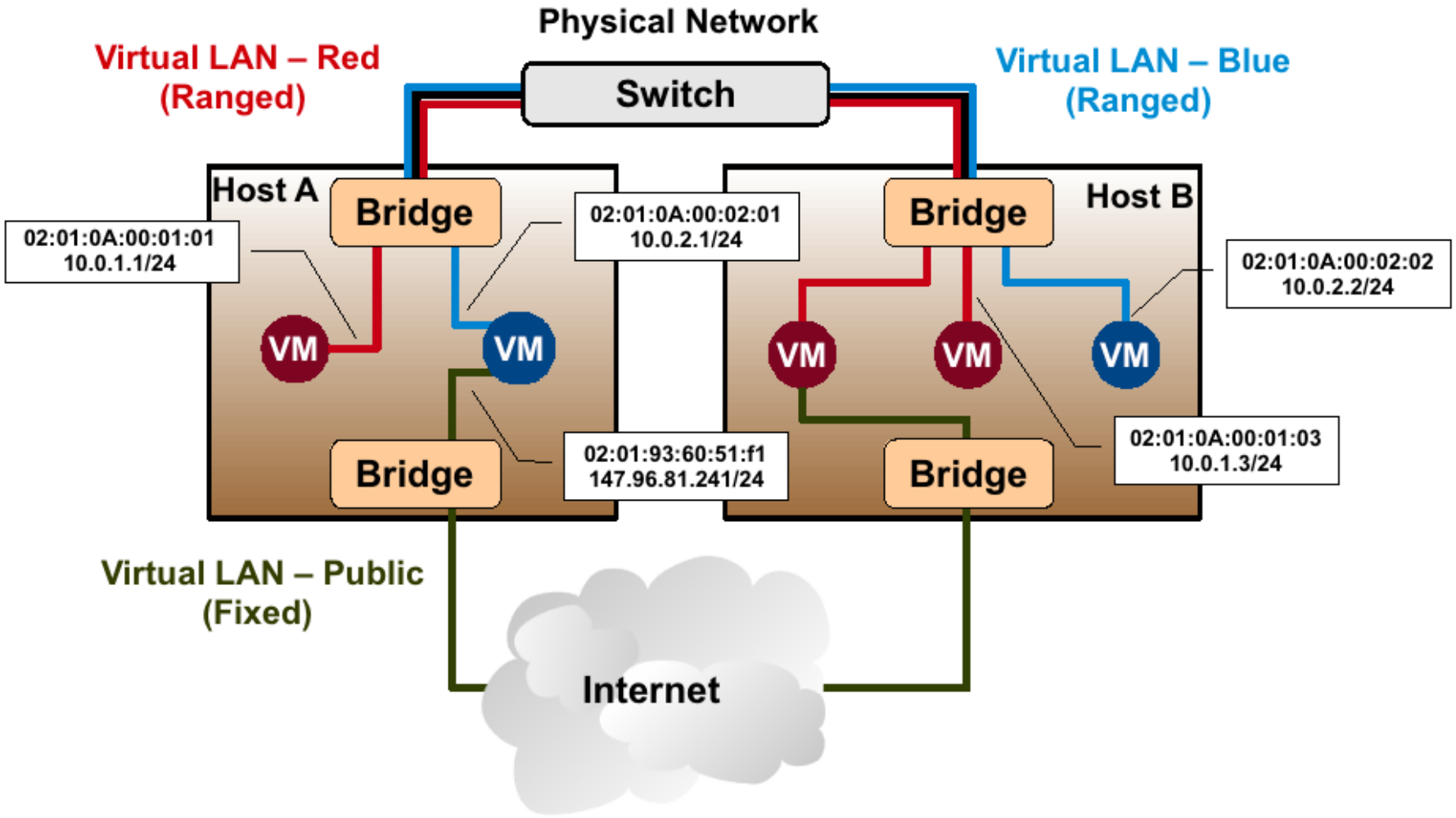
Virtual machines and their images are represented as block devices (just like a disk)

Networking for private clouds

- OpenNebula management operations use ssh connections
- ***Image traffic,*** may require the movement of heavy files (VM images, checkpoints). Dedicated storage links may be a good idea
- ***VM demands,*** consider the typical requirements of your VMs. Several NICs to support the VM traffic may be a good idea
- OpenNebula relies on bridge networking for the VMs



Example network setup in a private cloud



Virtual machines

VM Description

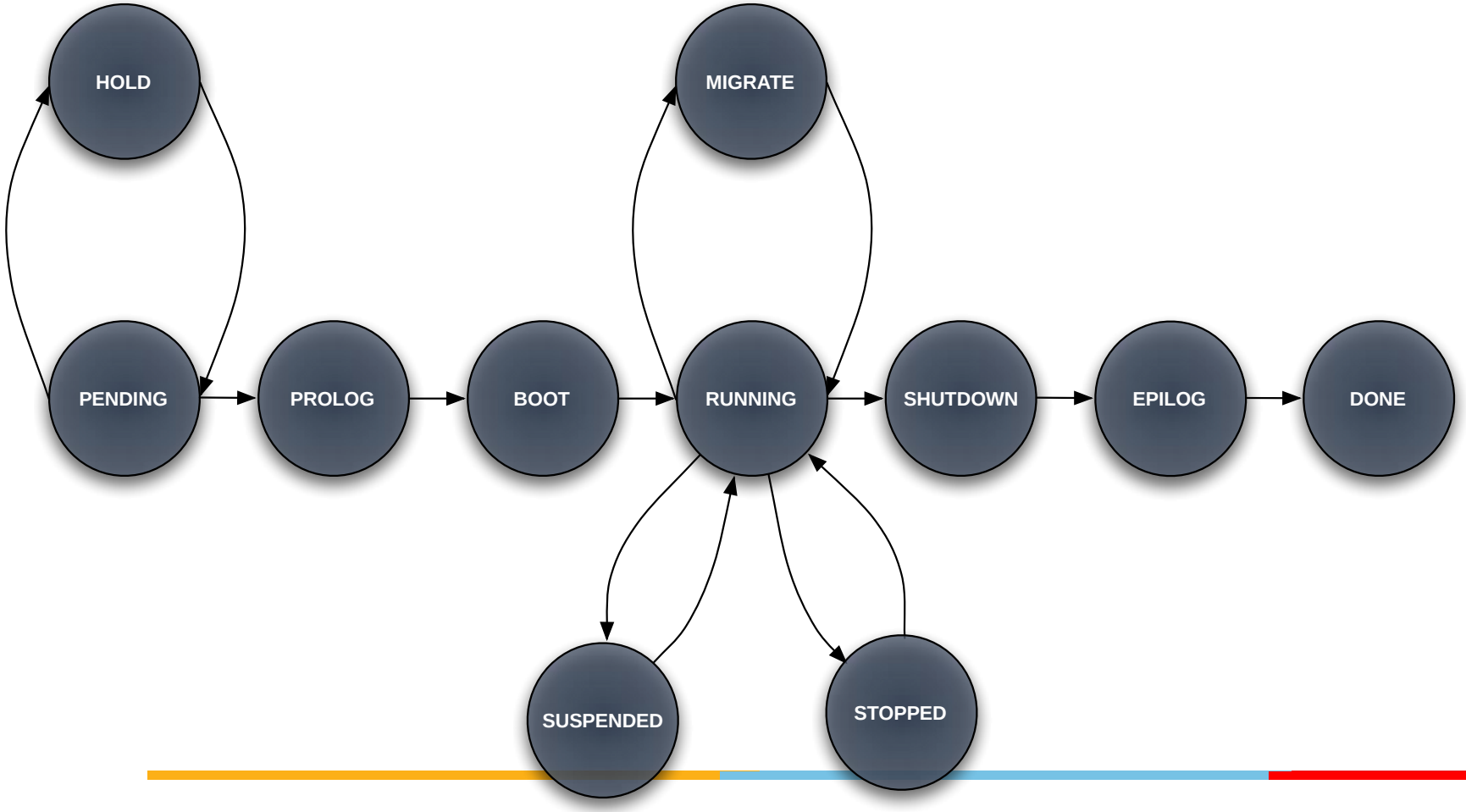
Option	Description
NAME	• Name that the VM will get for description purposes.
CPU	• Percentage of CPU divided by 100 required for the Virtual Machine.
OS (KERNEL, INITRD)	• Path of the kernel and initrd files to boot from.
DISK (SOURCE, TARGET, CLONE, TYPE)	• Description of a disk image to attach to the VM.
NIC (NETWORK)	• Definition of a virtual network the VM will be attached to.

Multiple disk an network interfaces can be specified just adding more disk/nic statements.

To create swap images you can specify TYPE=swap, SIZE=<*size in MB*>.

By default disk images are cloned, if you do not want that to happen CLONE=no can be specified and the VM will attach the original image.

VM States overview



Pending state

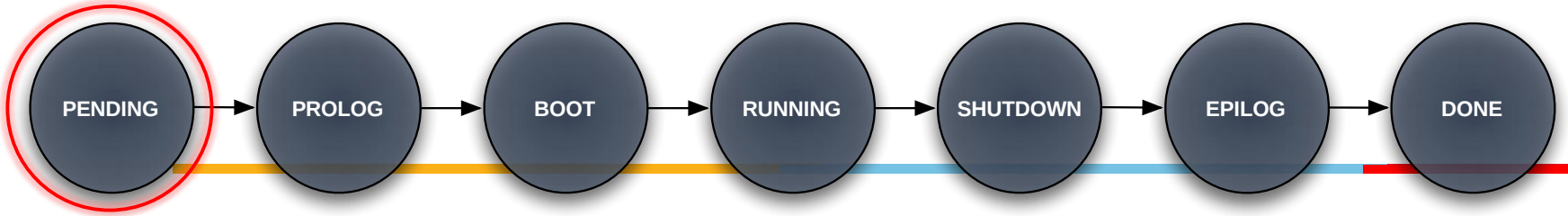
After submitting a VM description to ONE it is added to the database and its state is set to PENDING.

In this state IP and MAC addresses are also chosen if they are not explicitly defined.

The scheduler awakes every 30 seconds and looks for VM descriptions in PENDING state and searches for a physical node that meets its requirements. Then a deploy XML-RPC message is sent to *oned* to make it run in the selected node.

Deployment can be also made manually using the Command Line Interface:

⇒ onevm deploy <vmid> <hostid>

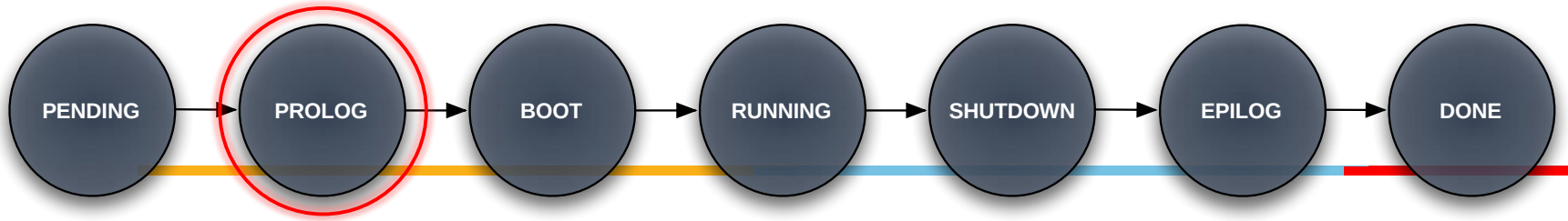


Prolog state

In PROLOG state the Transfer Driver prepares the images to be used by the VM.

Transfer actions:

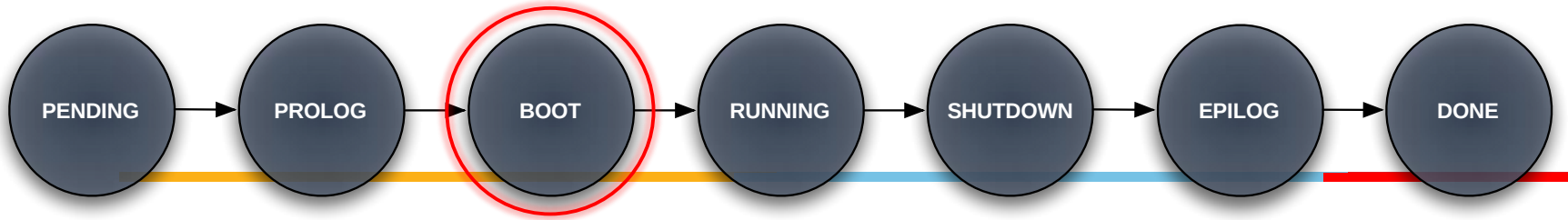
- **CLONE**: Makes a copy of a disk image file to be used by the VM. If Clone option for that file is set to false and the Transfer Driver is configured for NFS then a symbolic link is created.
- **MKSWAP**: Creates a swap disk image on the fly to be used by the VM if it is specified in the VM description.



Boot state

In this state a deployment file specific for the virtualization technology configured for the physical host is generated using the information provided in the VM description file. Then Virtual Machine Driver sends deploy command to the virtual host to start the VM.

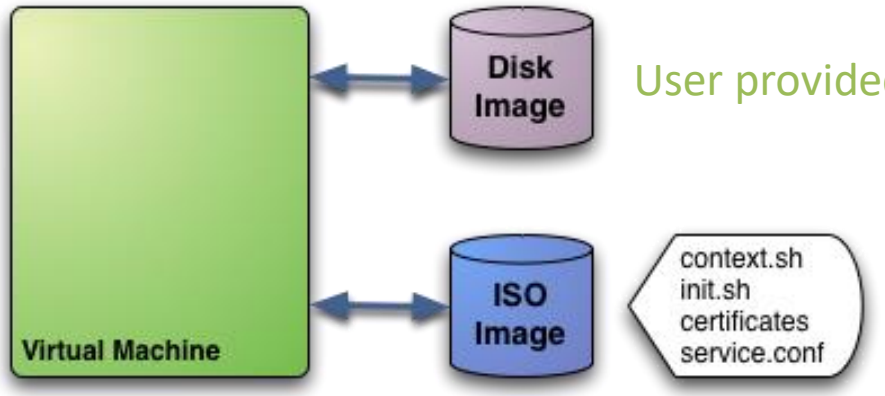
The VM will be in this state until deployment finishes or fails.



Contextualization

The ISO image has the contextualization for that VM:

- **context.sh**: contains configuration variables
- **init.sh**: script called by VM at start to configure specific services
- **certificates**: directory that contains certificates for some service
- **service.conf**: service configuration



User provided, OpenNebula provided contextualization info

Running and Shutdown states

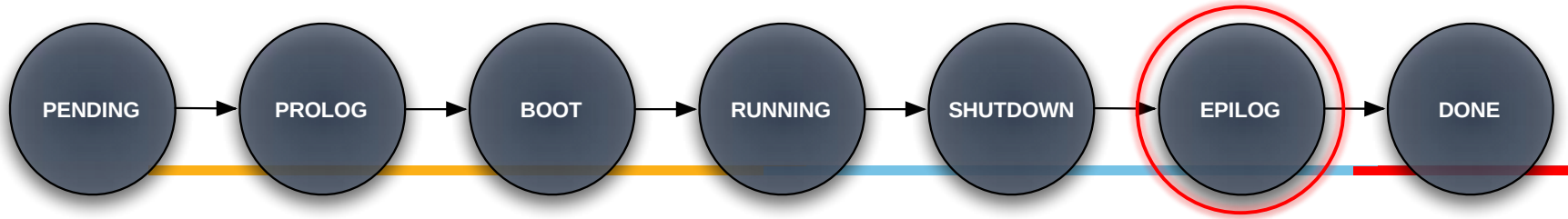
While the VM is in RUNNING state it will be periodically polled to get its consumption and state.

In SHUTDOWN state Virtual Machine Driver will send the shutdown command to the underlying virtual infrastructure.

Epilog state

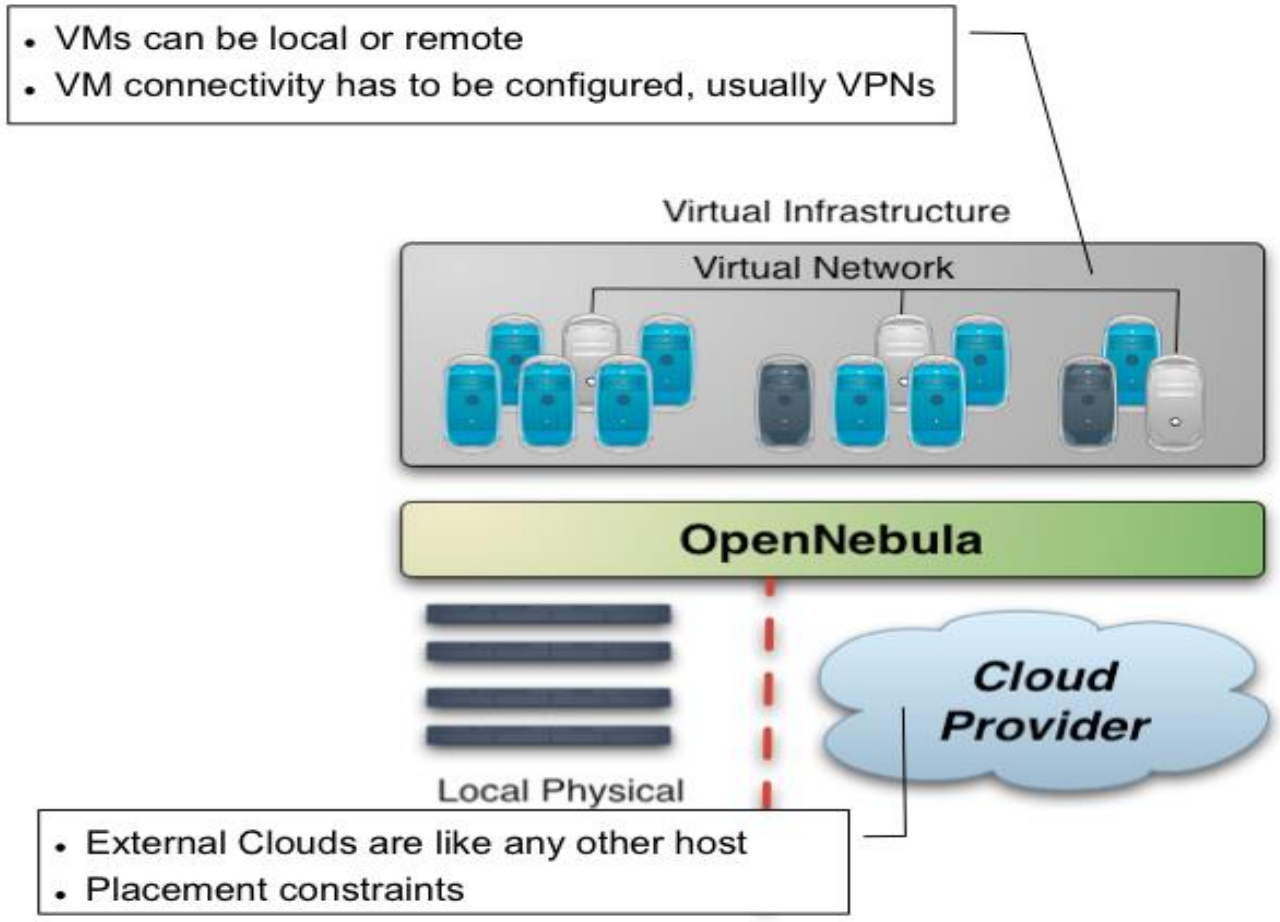
In EPILOG state the Transfer Manager Driver is called again to perform this actions:

- Copy back the images that have ***SAVE***=*yes* option.
- Delete images that were cloned or generated by **MKSWAP**.



Hybrid cloud

Overview


VMs can be local or remote
VM connectivity has to be configured, usually VPNs
Virtual Infrastructure
Virtual Network
OpenNebula
Cloud Provider
Local Physical
External Clouds are like any other host
Placement constraints

Making an Amazon EC2 hybrid

Amazon EC2 cloud is managed by OpenNebula as any other cluster node

- You can use several accounts by adding a driver for each account (use the arguments attribute, -k and -c options). Then create a host that uses the driver
- You can use multiple EC2 zones, add a driver for each zone (use the arguments attribute, -u option), and a host that uses that driver
- You can limit the use of EC2 instances by modifying the IM file

Using an EC2 hybrid cloud

Virtual Machines can be instantiated locally or in EC2

The VM template must provide a description for both instantiation methods.

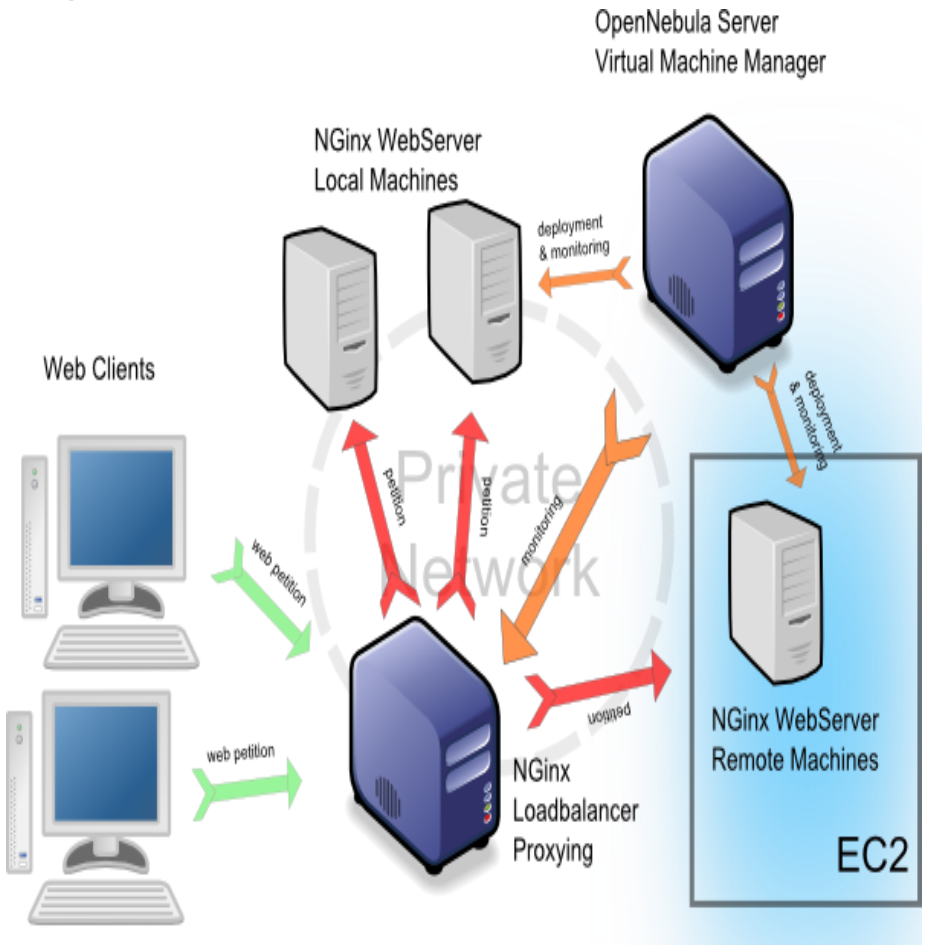
The EC2 counterpart of your VM (AMI_ID) must be available for the driver account

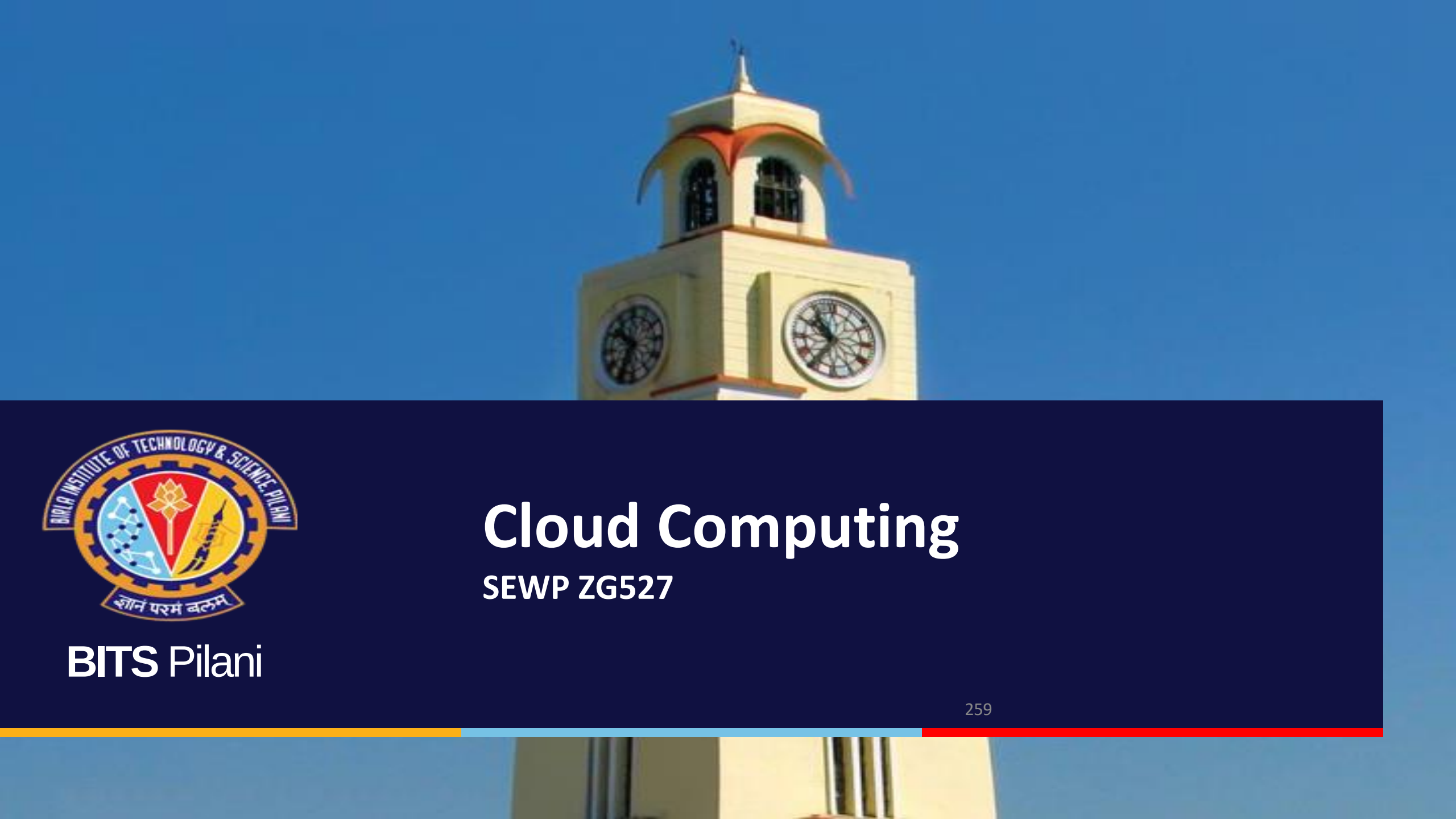
The EC2 VM template attribute should describe not only the VM's properties but the contact details of the external cloud provider

Hybrid cloud Use Case

On-demand Scaling of Computing Clusters

- On-demand Scaling of Web Servers
- Elastic execution of the NGinx web server
- The capacity of the elastic web application can be dynamically increased or decreased by adding or removing NGinx instances





BITS Pilani

Cloud Computing

SEWP ZG527

High Availability Requirements in Cloud

- **What is meant by High Availability** ?
  - Defined by Service Level Agreement (SLA):
  - HA goal is to meet SLA requirement
  - Balance between the availability and implementation cost
  - SLA: for example, 99.95%, annual 4 hrs 22 minutes downtime
    Downtime window: first Saturday: 8pm-10pm every quarter
- **Cases impacting system availability**:
  - Service outage by unplanned downtime:
    hardware or software failure, human error
  - Service disruption by planned downtime:
    hardware/software upgrade, patching and migration from old system to new system
  - Service performance degrade: violate performance SLA
    for example, 99% transactions finished in a 2 seconds window

High Availability Requirements in Cloud

- **High Availability SLA in Cloud Environment**
  - Consolidation of databases and applications in Cloud
  - Applications share the same cloud infrastructure
  - Great business impact due to the cloud infrastructure downtime
  - Applications may have different SLAs for different business:
    - Private cloud serves applications from different time zones
    - Public cloud serves different customers applications
    - Very difficult to find downtime to meet all the SLAs
- **Architect a High Availability Cloud Infrastructure**
  - How to design high available infrastructure for cloud
  - Architect hardware infrastructure to reduce unplanned outage
  - Design system architecture to minimize the planned/unplanned outage
  - Use configuration and implementation best practices for HA
  - Minimize downtime during system migration
  - Establish the pre-active real time monitoring system

Steps to achieve high availabilty

Build for server failure

Build for zone failure

Build for Cloud failure

Automating and testing

In the early days of web-application deployment, **performance of the application** at peak load was a single important criterion for provisioning server resources.

Provisioning in those days involved deciding hardware configuration, determining the number of physical machines, and acquiring them upfront so that the overall business objectives could be achieved.

The web applications were **hosted** on these dedicated individual servers within enterprises' own server rooms. These web applications were used to provide different kinds of e-services to various clients.

Typically, the **service-level objectives** (SLOs) for these applications were response time and throughput of the application end-user requests.

The capacity buildup was to cater to the estimated peak load experienced by the application. The activity of determining the number of servers and their capacity that could satisfactorily serve the application end-user requests at peak loads is called capacity planning

Key aspects of SLA

Due to the increasing **complexity of managing the hugh Data centres**, enterprises started outsourcing the application hosting to the infrastructure providers. They would procure the hardware and make it available for application hosting.

It necessitated the enterprises to enter into a **legal agreement with the infrastructure service providers** to guarantee a minimum quality of service (QoS).

Typically, the **QoS parameters** are related to the availability of the system CPU, data storage, and network for efficient execution of the application at peak loads.

This legal agreement is known as the service-level agreement (SLA)

Key aspects of SLA

For example, one SLA may state that the application's server machine will be available for 99.9% of the key business hours of the application's end users, also called core time, and 85% of the non-core time.

Another SLA may state that the service provider would respond to a reported issue in less than 10 minutes during the core time, but would respond in one hour during non-core time.

These SLAs are known as the infrastructure SLAs, and the infrastructure service providers are known as **Application Service Providers (ASPs)**

Key aspects of SLA

The dedicated hosting practice resulted in **massive redundancies** within the ASP's data centers **due to the underutilization of many of their servers**. This is because the applications were not fully utilizing their servers' capacity at nonpeak loads.

To **reduce the redundancies** and increase the server utilization in data centers, ASPs started co-hosting applications with complementary workload patterns.

Co-hosting of applications means **deploying more than one application on a single server**. This led to further cost advantage for both the ASPs and enterprises.

Key aspects of SLA

However, newer challenges such as application performance isolation and **security guarantees** emerged and needed to be addressed. Performance isolation implies that one application should not steal the resources being utilized by other co-located applications.

Hence, appropriate measures are needed to **guarantee security and performance isolation**. These challenges prevented ASPs from fully realizing the benefits of co-hosting.

Virtualization technologies have been proposed to overcome the above challenges. The ASPs could exploit the containerization features of virtualization technologies to provide performance isolation and guarantee data security to different co-hosted applications

Key aspects of SLA

Adoption of virtualization technologies required ASPs to get more detailed insight into the application runtime characteristics with high accuracy. Based on these characteristics, **ASPs can allocate system resources more efficiently to these applications on-demand**, so that application-level metrics can be monitored and met effectively.

These metrics are request rates and response times. Therefore, different SLAs than the infrastructure SLAs are required. These SLAs are called application SLAs. These service providers are known as **Managed Service Providers (MSP)** because the service providers were responsible for managing the application availability too.

Key aspects of SLA

To **fulfill the SLOs** mentioned in the application SLA and also make their IT infrastructure elastic, an in-depth understanding of the application's behavior is required for the MSPs. Elasticity implies progressively scaling up the IT infrastructure to take the increasing load of an application. The customer is billed based on their application usage of infrastructure resources for a given period only. The infrastructure can be augmented by procuring resources dynamically from multiple sources, including other MSPs, if resources are scarce at their data centers. This kind of new hosting infrastructure is called cloud platform.

Key aspects of SLA

Traditionally, load balancing techniques and admission control mechanisms have been used to **provide guaranteed quality of service (QoS)** for hosted web applications. These mechanisms can be viewed as the first attempt towards managing the SLOs.

Now it is also possible for a **customer and the service provider to mutually agree upon a set of SLAs** with different performance and cost structure rather than a single SLA. The customer has the flexibility to choose any of the agreed SLAs from the available offerings. At runtime, the customer can switch between the different SLAs.

Key Components of a Service-Level Agreement

Service Level Parameter

Describes an observable property of a service whose value is measurable. **Metrics** These are definitions of values of service properties that are measured from a service providing system or computed from other metrics and constants.

Metrics are the key instrument to describe exactly what SLA parameters mean by specifying how to measure or compute the parameter values.

Function A function specifies how to compute a metric's value from the values of other metrics and constants. Functions are central to describing exactly how SLA parameters are computed from resource metrics.

Measurement directives These specify how to measure a metric.

Key aspects of SLA

Key Contractual Elements of an Infrastructural SLA Hardware availability 99% uptime in a calendar month Power availability 99.99% of the time in a calendar month

Data center network availability 99.99% of the time in a calendar month

Backbone network availability 99.999% of the time in a calendar month

Service credit for unavailability Refund of service credit prorated on downtime period

Outage notification guarantee Notification of customer within 1 hr of complete downtime

Internet latency guarantee When latency is measured at 5 min intervals to an upstream provider, the average doesn't exceed 60 msec

Packet loss guarantee Shall not exceed 1% in a calendar month

Key contractual components of an application SLA

Service level parameter metric Web site response time (e.g., max of 3.5 sec per user request)

Latency of web server (WS) (e.g., max of 0.2 sec per request)

Latency of DB (e.g., max of 0.5 sec per query)Function

Average latency of WS (latency of web server 1+latency of web server 2) /2

Web site response time Average latency of web server+ latency of database Measurement directive

DB latency available via http://mgmtserver/em/latency WS latency available via http://mgmtserver/ws/instanceno/ latency

Service level objective Service assurance

web site latency , 1 sec when concurrent connection , 1000

Penalty 1000 USD for every minute while the SLO was breached

Key aspects of SLA

Each SLA goes through a sequence of steps starting from identification of terms and conditions, activation and monitoring of the stated terms and conditions, and eventual termination of contract once the hosting relationship ceases to exist.

Such a sequence of steps is called **SLA life cycle** and consists of the following five phases:

1. Contract definition
2. Publishing and discovery
3. Negotiation
4. Operationalization
5. De-commissioning

Key aspects of SLA

.Some of the parameters

The SLA class (Platinum, Gold, Silver, etc.) to which the application belongs to.

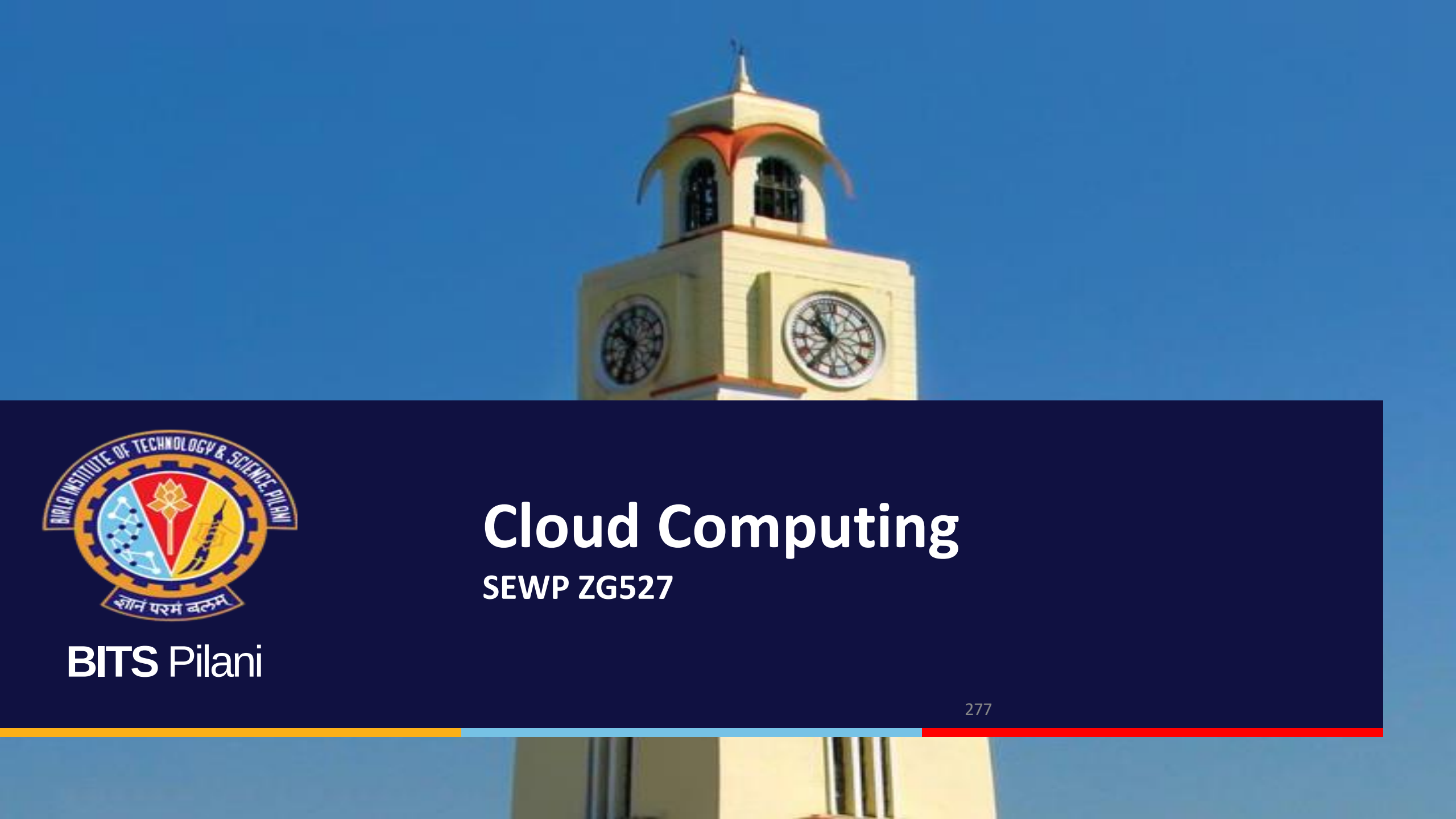
The amount of penalty associated with SLA breach. Whether the application is at the threshold of breaching the SLA.

Whether the application has already breached the SLA.

The number of applications belonging to the same customer that has breached SLA.

The number of applications belonging to the same customer about to breach SLA

. The type of action to be performed to rectify the situation.

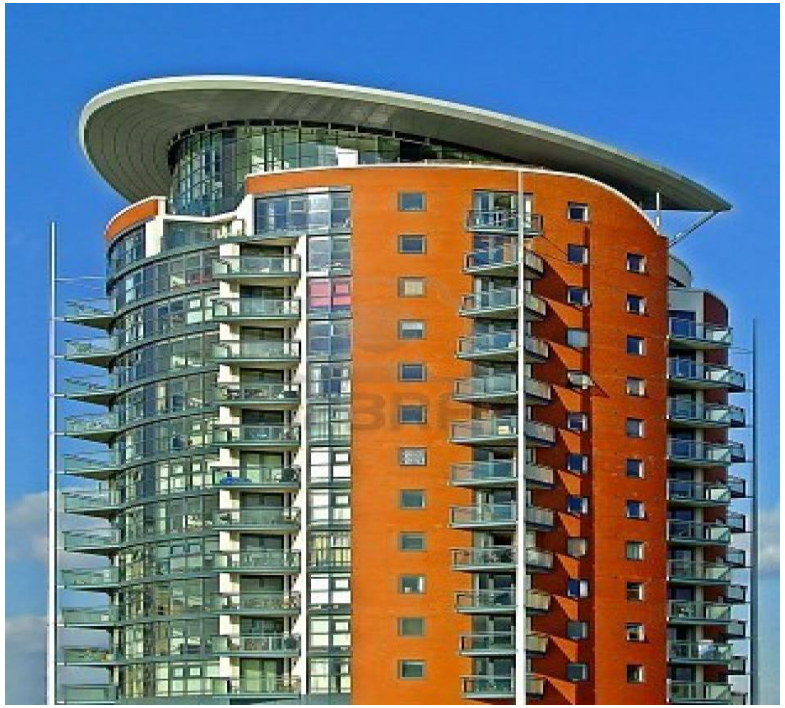


BITS Pilani

Cloud Computing

SEWP ZG527

Multitenancy – What is it?

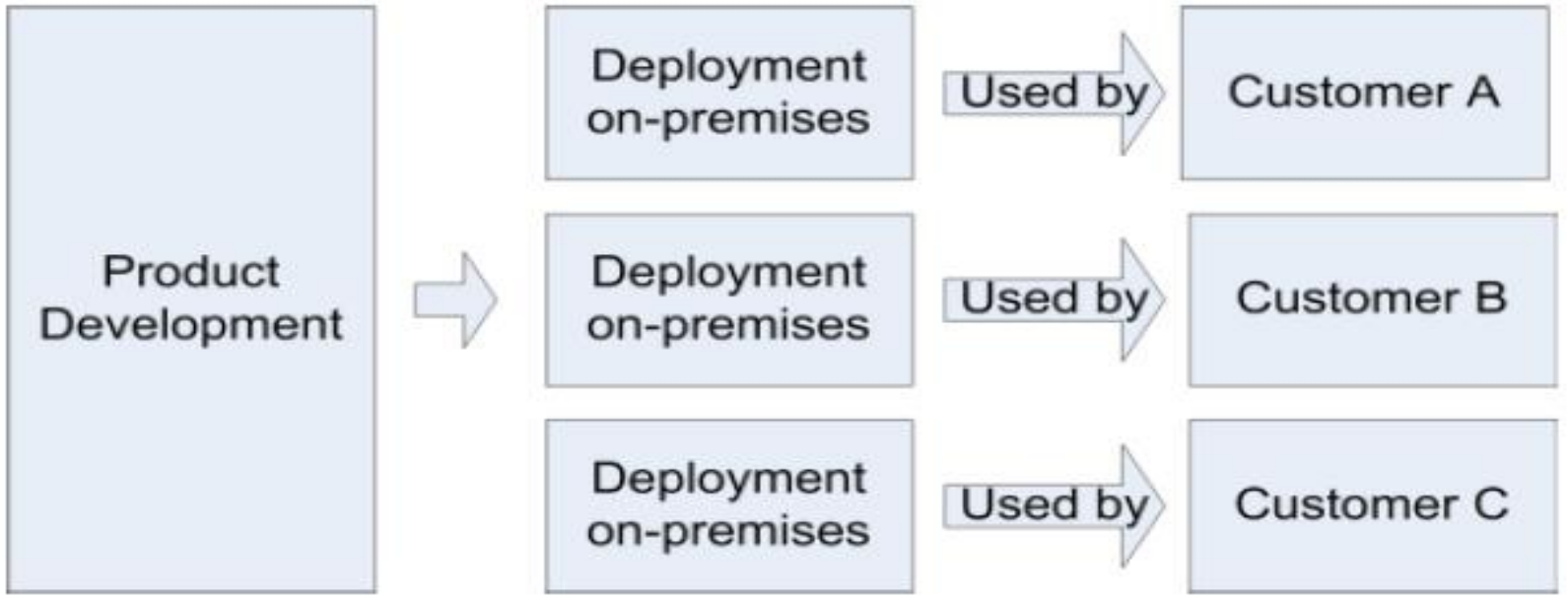


Pros and Cons

Effective use of land	**-**	**+**
Privacy	**+**	**-**
Infrastructure sharing	**-**	**+**
Maintenance cost sharing	**-**	**+**
Freedom	**+**	**-**

House: Privacy and freedom
Apartment: Cost efficiency

Traditional Deployment Model



Multitenancy – Introduction

- Multi-tenancy is an architecture in which a single instance of a software application serves multiple customers. Each customer is called a tenant. Tenants may be given the ability to customize some parts of the application, such as color of the user interface (UI) or business rules, but they cannot customize the application's code.
- A software-as-a-service (SaaS) provider, for example, can run one instance of its application on one instance of a database and provide web access to multiple customers. In such a scenario, each tenant's data is isolated and remains invisible to other tenants.

- Multi-tenancy is an architectural pattern
- A single instance of the software is run on the service provider's infrastructure
- Multiple tenants access the same instance.
- In contrast to the multi-user model, multi-tenancy requires customizing the single instance according to the multi-faceted requirements of many tenants.

Multitenancy – key aspects

A Multi-tenants application lets customers (tenants) share the same hardware resources, by offering them one shared application and database instance ,while allowing them to configure the application to fit there needs as if it runs on dedicated environment.

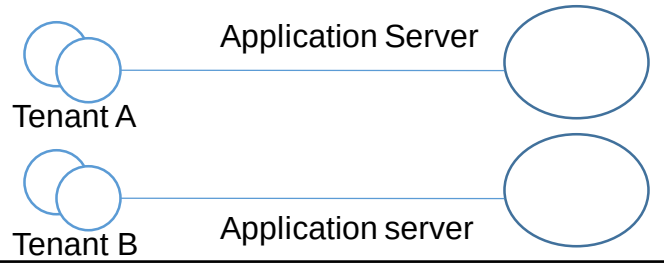
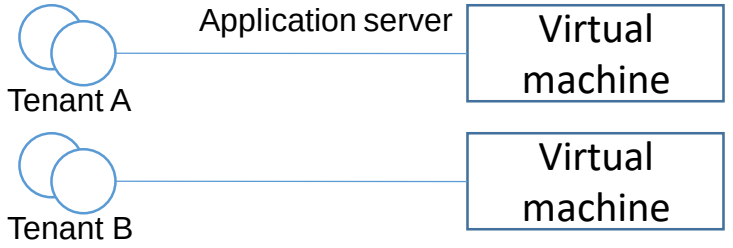
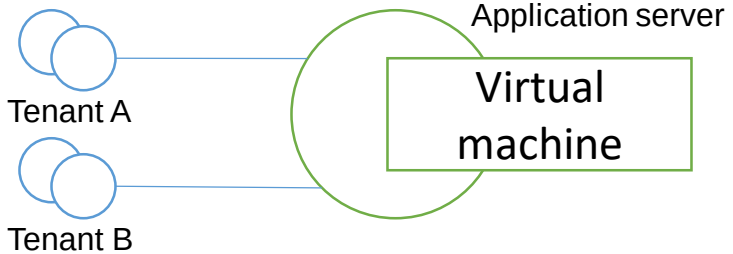
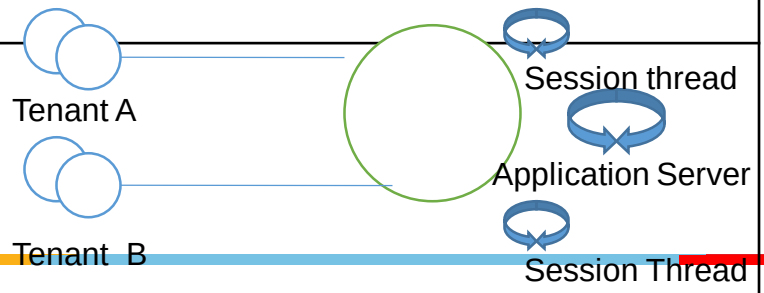
These definition focus on what we believe to be the key aspects of multi tenancy:

1.The ability of the application to share hardware resources.

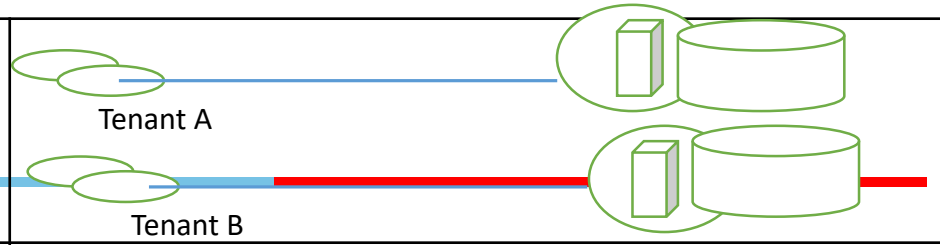
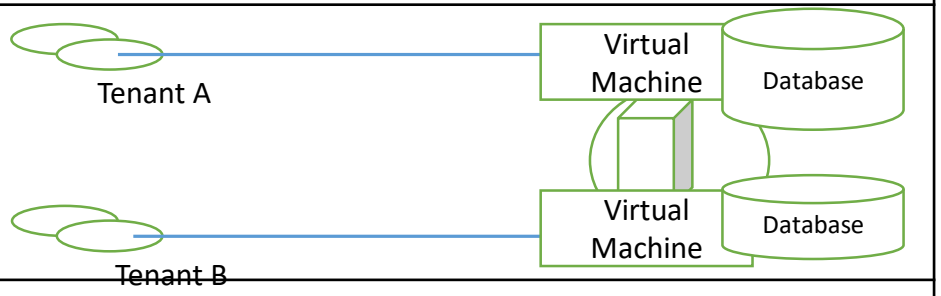
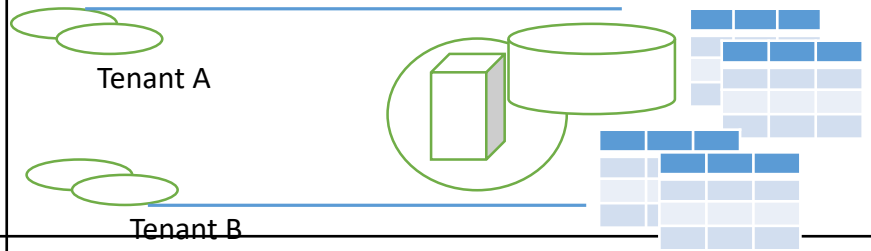
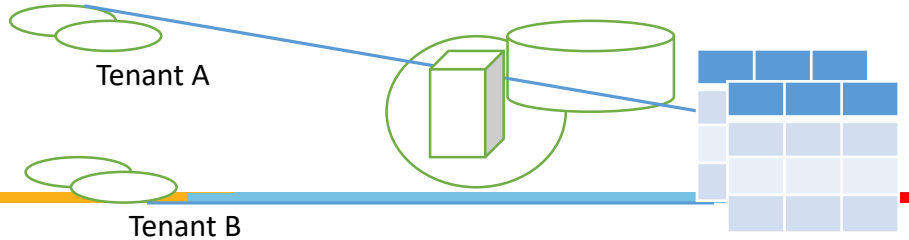
2.The offering of a high degree of configurability of the software.

3.The architectural approach in which the tenants make use of a single application and database instance.

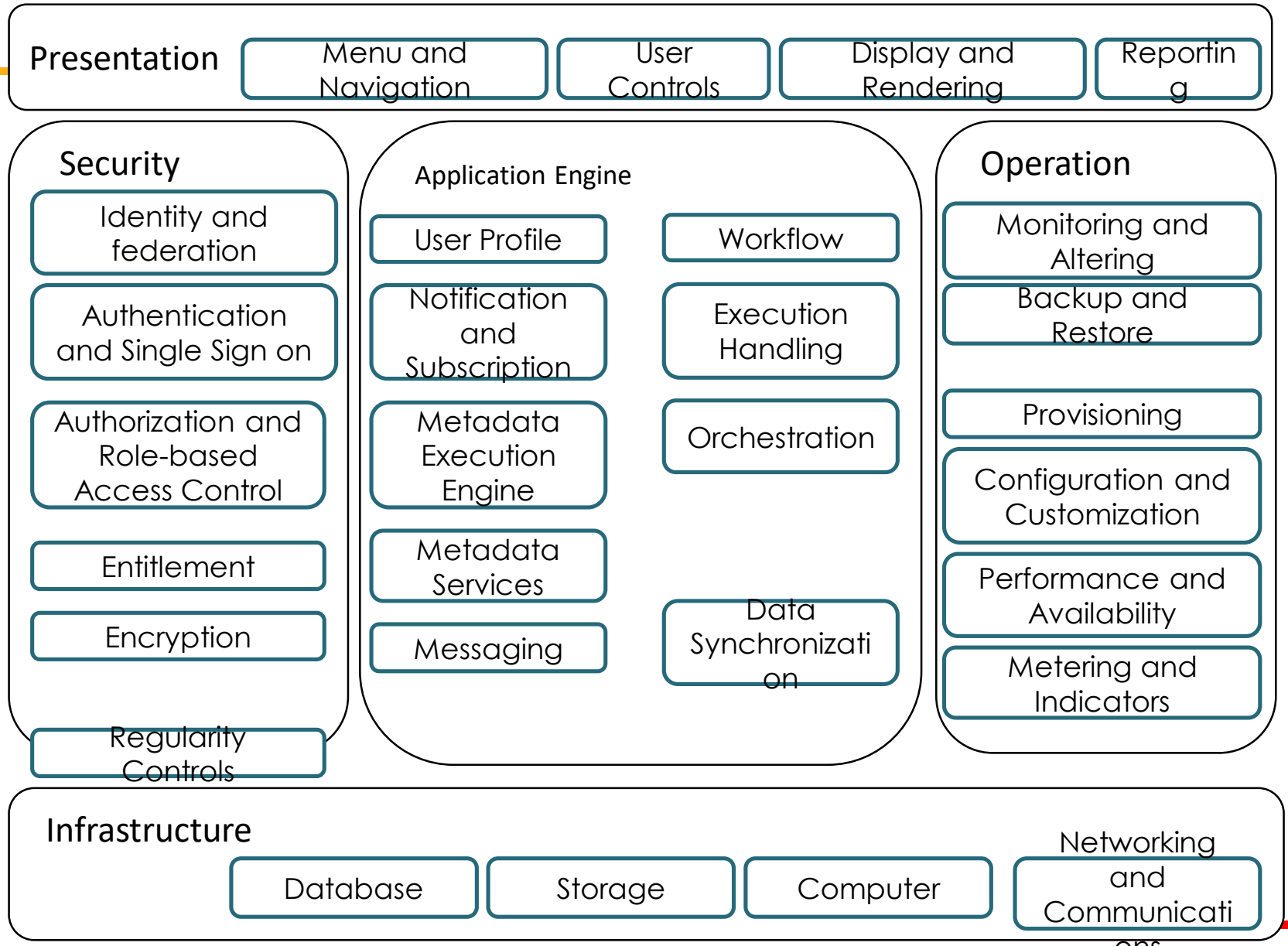
Multi-tenants Deployment Modes for Application Server

Fully isolated Application server Each tenant accesses an application server running on a dedicated servers.	
Virtualized Application Server Each tenant accesses a dedicated application running on a separate virtual machine.	
Shared Virtual Server Each tenant accesses a dedicated application server running on a shared virtual machine.	
Shared Application Server The tenant shared the application server and access application resources through separate session or threads	

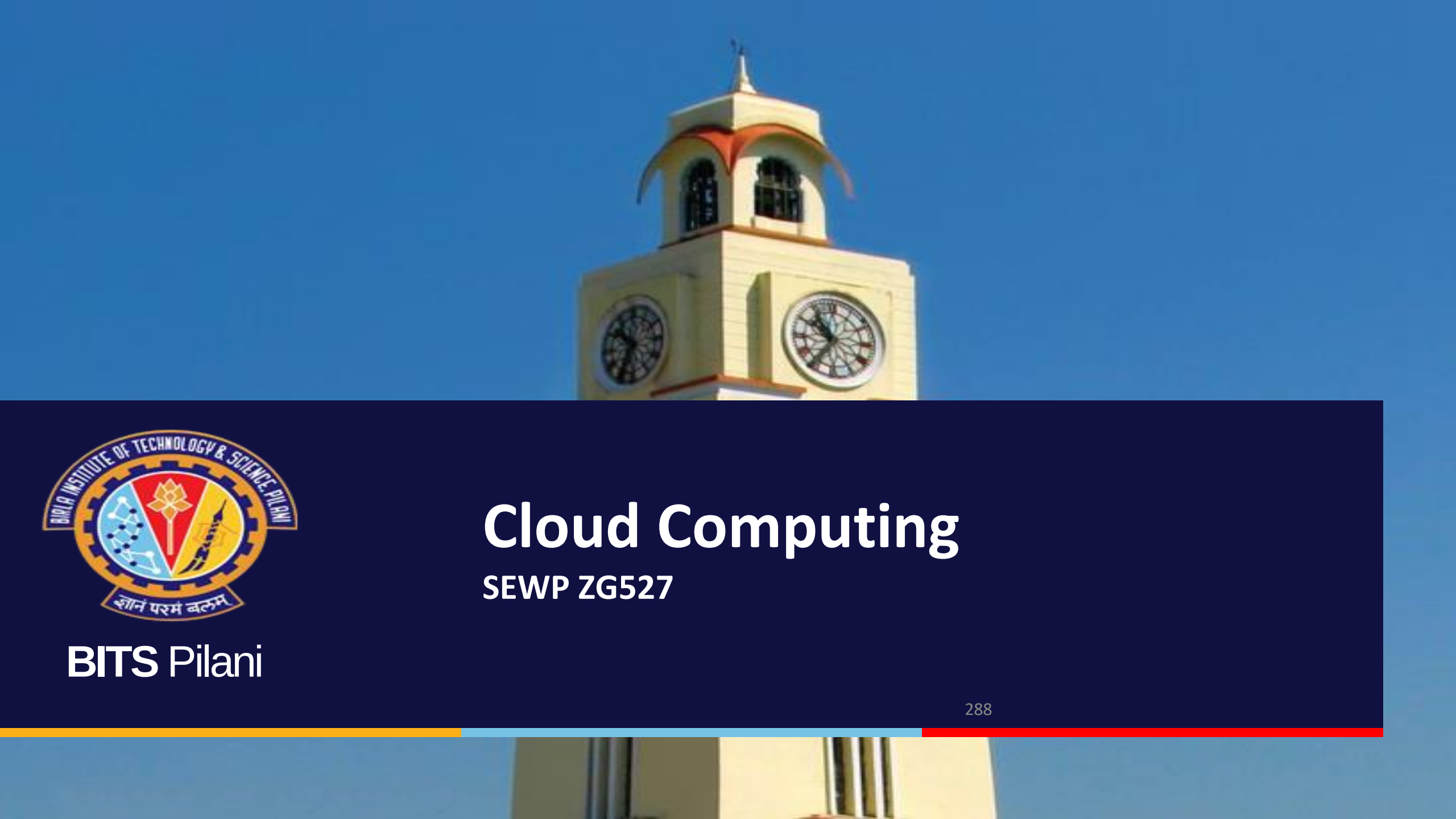
Multi-tenants Deployment Modes in Data Centers

Fully isolated data center **The tenants do not share any data center resources**	
Virtualized servers **The tenants share the same host but access different databases running on separate virtual machines**	
Shared Server **The tenants share the same server (Hostname or IP) but access different databases**	
Shared Database **The tenants share the same server and database (shared or different ports) but access different schema(tables)**	
Shared Schema **The tenants share the same server, database and schema (tables). The irrespective data is segregated by key and rows**	

Conceptual framework of Software as a Service



Thank you



BITS Pilani

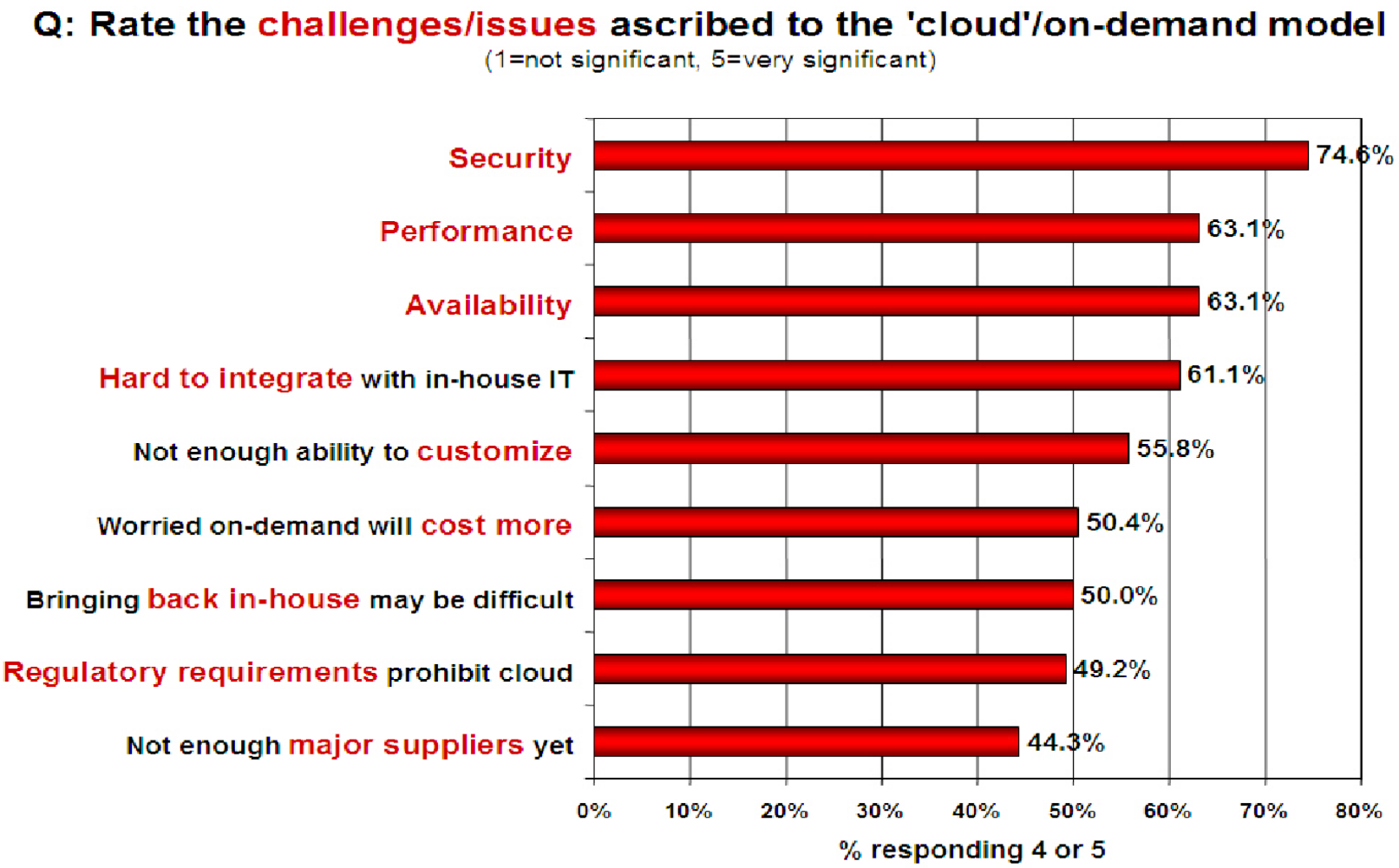
Cloud Computing

SEWP ZG527

If cloud computing is so great, why isn't everyone doing it?

- The cloud acts as a big black box, nothing inside the cloud is visible to the clients
- Clients have no idea or control over what happens inside a cloud
- Even if the cloud provider is honest, it can have malicious system admins who can tamper with the VMs and violate confidentiality and integrity
- Clouds are still subject to traditional data confidentiality, integrity, availability, and privacy issues, plus some additional attacks

Companies are still afraid to use clouds



Source: IDC Enterprise Panel, August 2008 n=244

Cloud Security Issues

- Most security problems stem from:
  - Loss of Control
    - Take back control
      - Data and apps may still need to be on the cloud
      - But can they be managed in some way by the consumer?
  - Lack of trust
    - Increase trust (mechanisms)
      - Technology
      - Policy, regulation
      - Contracts (incentives): topic of a future talk
  - Multi-tenancy
    - Private cloud
      - Takes away the reasons to use a cloud in the first place
    - VPC: its still not a separate system
    - Strong separation
- These problems exist mainly in 3rd party management models
  - Self-managed clouds still have security issues, but not related to above

Consumer's loss of control

- Data, applications, resources are located with provider
- User identity management is handled by the cloud
- User access control rules, security policies and enforcement are managed by the cloud provider
- Consumer relies on provider to ensure
  - Data security and privacy
  - Resource availability
  - Monitoring and repairing of services/resources

- Conflict between tenants' opposing goals
  - Tenants share a pool of resources and have opposing goals
- How does multi-tenancy deal with conflict of interest?
  - Can tenants get along together and 'play nicely' ?
  - If they can't, can we isolate them?
- How to provide separation between tenants?
- Cloud Computing brings new threats

Multiple independent users share the same physical infrastructure

Thus an attacker can legitimately be in the same physical machine as the target

Taxonomy of Fear

- Confidentiality
  - Fear of loss of control over data
    - Will the sensitive data stored on a cloud remain confidential?
    - Will cloud compromises leak confidential client data
  - Will the cloud provider itself be honest and won't peek into the data?
- Integrity
  - How do I know that the cloud provider is doing the computations correctly?
  - How do I ensure that the cloud provider really stored my data without tampering with it?

Taxonomy of Fear

Availability

- Will critical systems go down at the client, if the provider is attacked in a Denial of Service attack?
- What happens if cloud provider goes out of business?
- Would cloud scale well-enough?
- Often-voiced concern
- Although cloud providers argue their downtime compares well with cloud user's own data centers

Taxonomy of Fear

- Privacy issues raised via massive data mining
  - Cloud now stores data from a lot of clients, and can run data mining algorithms to get large amounts of information on clients
- Increased attack surface
  - Entity outside the organization now stores and computes data, and so
  - Attackers can now target the communication link between cloud provider and client
  - Cloud provider employees can be phished

Taxonomy of Fear

- Audit-ability and forensics (out of control of data)
  - Difficult to audit data held outside organisation in a cloud
  - Forensics also made difficult since now clients don't maintain data locally
- Legal quagmire and transitive trust issues
  - Who is responsible for complying with regulations?
  - e.g., SOX, HIPAA, GLBA ?
  - If cloud provider subcontracts to third party clouds, will the data still be secure?

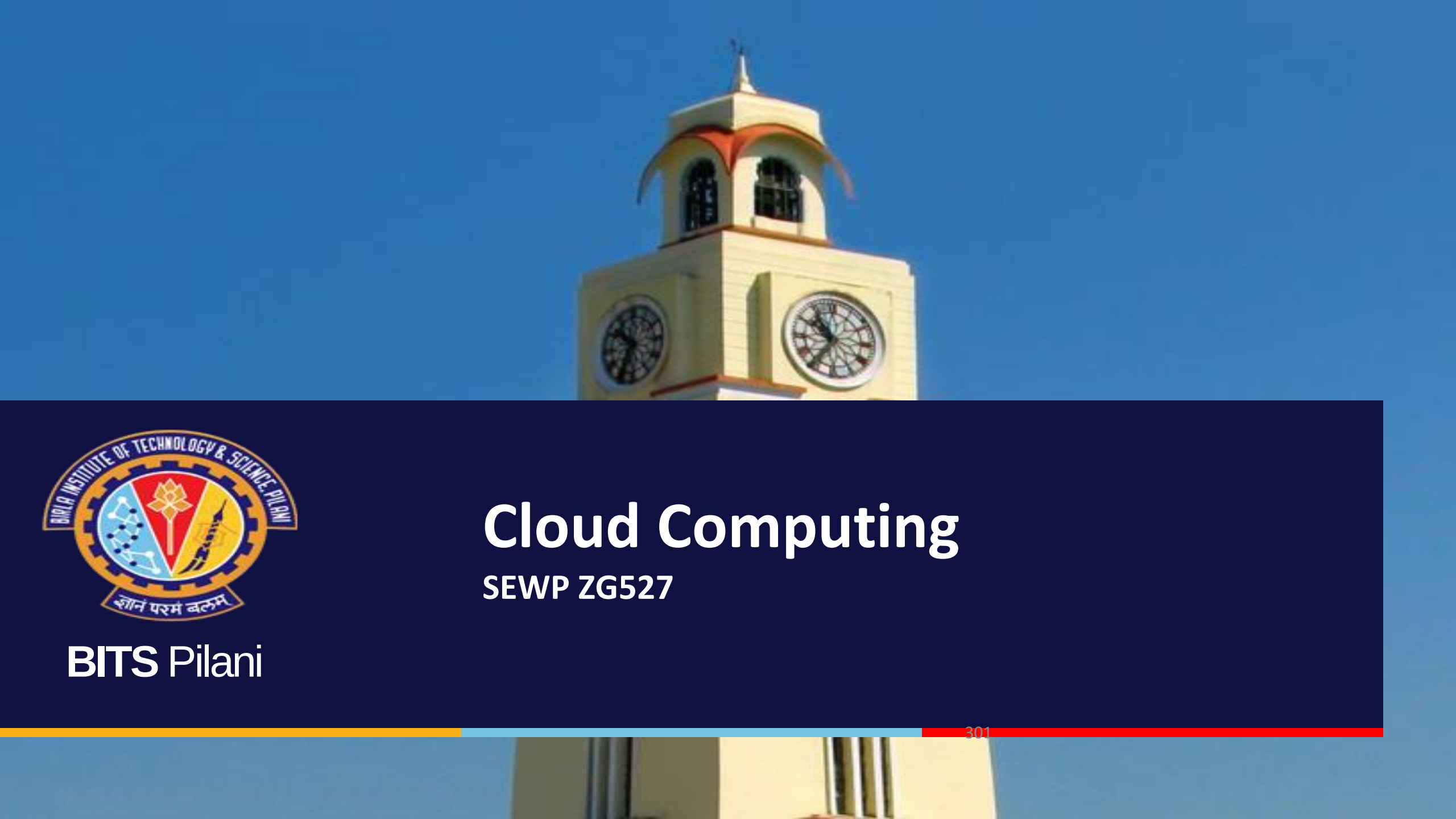
Threat Model

- A threat model helps in analysing a security problem, design mitigation strategies, and evaluate solutions
- Steps:
  - Identify attackers, assets, threats and other components
  - Rank the threats
  - Choose mitigation strategies
  - Build solutions based on the strategies

Threat Model

- Basic components
  - Attacker modelling
    - Choose what attacker to consider
      - insider vs. outsider?
      - single vs. collaborator?
    - Attacker motivation and capabilities
  - Attacker goals
  - Vulnerabilities / threats

Thank you

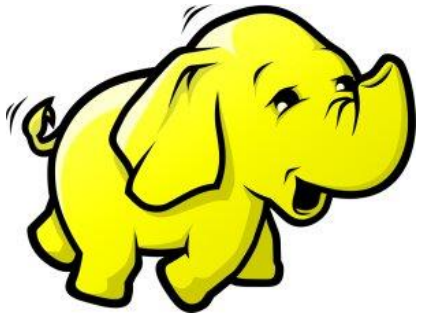


BITS Pilani

Cloud Computing

SEWP ZG527

Hadoo



Name of the elephant!!

Or have u thought of the following?

- How do “big bazaar/more/D’Mart” target promotions guaranteed to make you buy?

- How can Airtel(4G) increase Ad-campaign efficiency?

- What’s in your search? How is Google able to make such good predictions about your search?

- I have huge amount of data(nw sites) data(twitter) data(data(feeds) data(forums) ? What do I do with it?

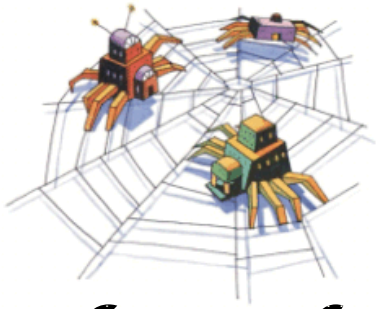
Wow, that's so much of DATA to process!!!

: exactly, and that what we call as "BIG Data"

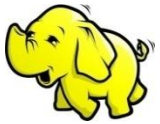
- Hadoop is one of the best-known cloud platforms for big data today
- It solves a specific class of data-crunching problems that frequently comes up in the domain of Internet computing and high-performance computing.
- Managing lots of information (growing by the day and doubling by year)
- Working with many new types of data (totally unstructured)

One of the research in the year 2012, Hadoop held the world record for the fastest system to sort large data (500 GB of data in59 sec and 100 terabytes of data in 68 seconds)

Designed to answer the question: **“How to process big data with reasonable cost and time?”**



Super, so tell me more about Hadoop, the data cruncher

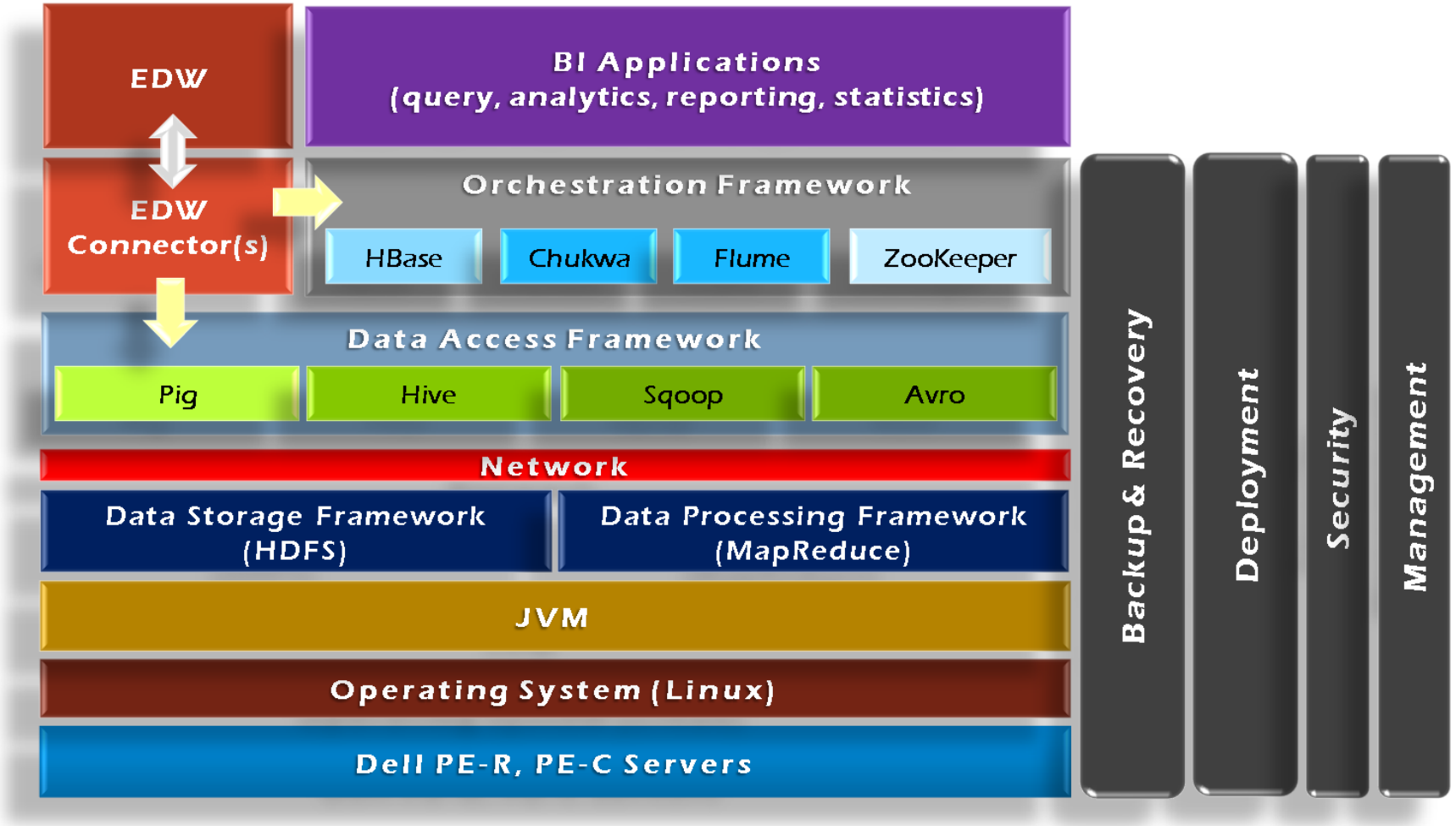


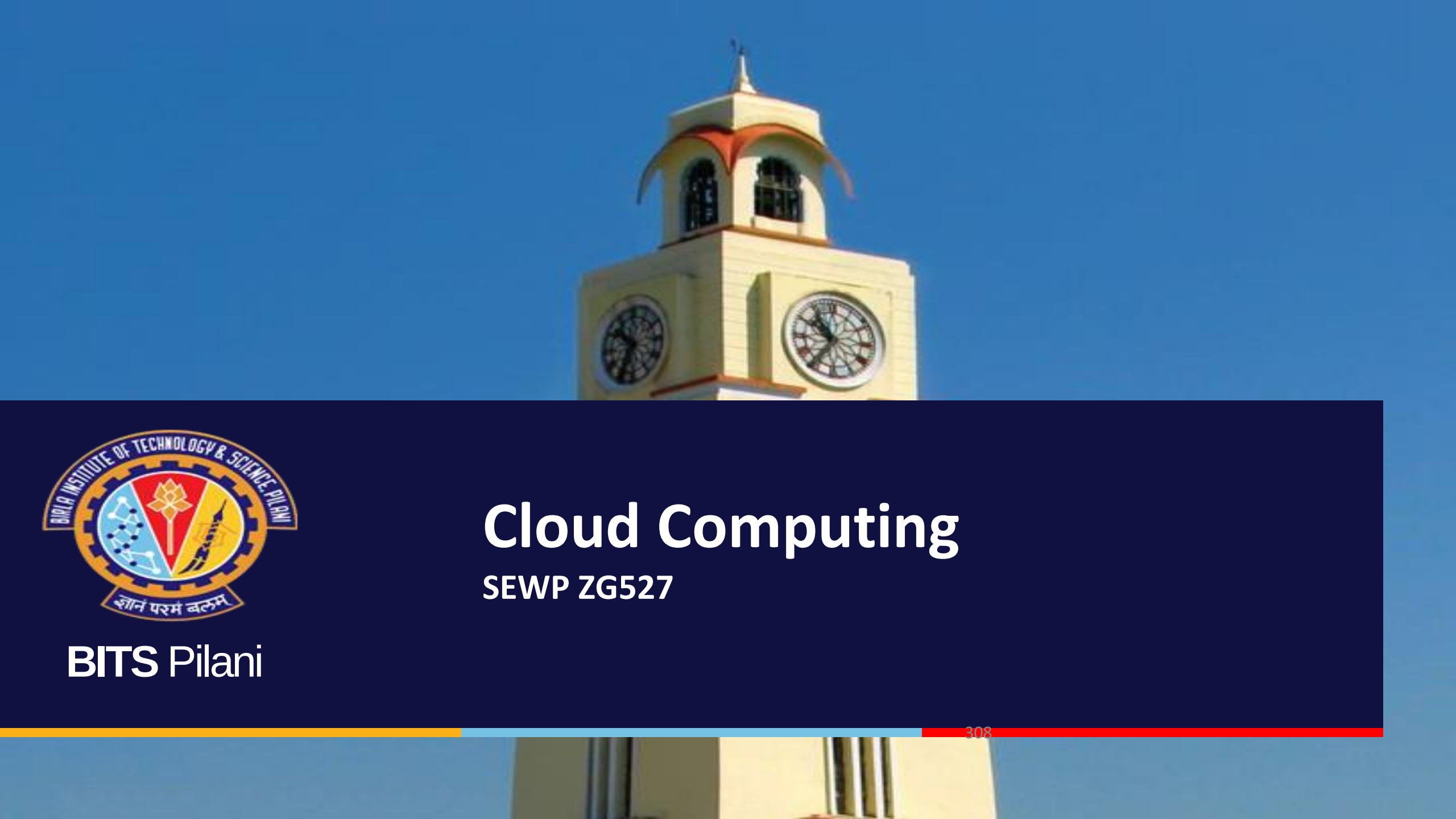
Okay, okay…. Sit tight,

Hadoop Features

- Hadoop is optimized for batch-processing applications, and scales to the number of CPUs available in the cluster
- Provides Framework for Massive parallel processing
- Programmer can focus on their program, and the framework takes care of the details of parallelization, fault-tolerance, locality optimization, load balancing
- **Paradigm shift:** In MapReduce programming model, computation goes to data rather than data coming to program. Processing takes place where data is.

Hadoop Framework Tools



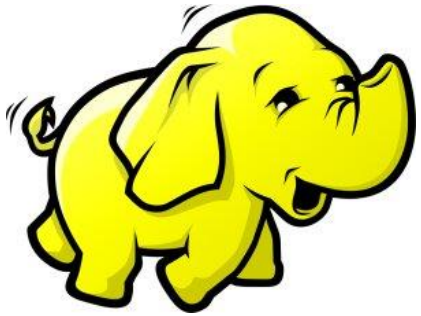


BITS Pilani

Cloud Computing

SEWP ZG527

Hadoo



Name of the elephant!!

Hadoop common Component

MapReduce – offline computing engine (Data Processing Framework)

HDFS – Hadoop Distributed file system (Data Storage Framework

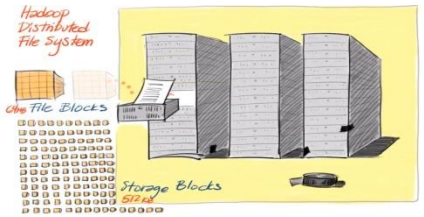
Frameworks like **Hbase**, **Pig** and **Hive** have been built on top of Hadoop.

- **Pig** is a dataflow language and execution environment over Hadoop.
- **Hbase** is a distributed key-value store which supports SQL-like queries similar to Google's BigTable
- **Hive** is a distributed data warehouse to manage data stored in the Hadoop File System.

MapReduce (Data Processing Framework)

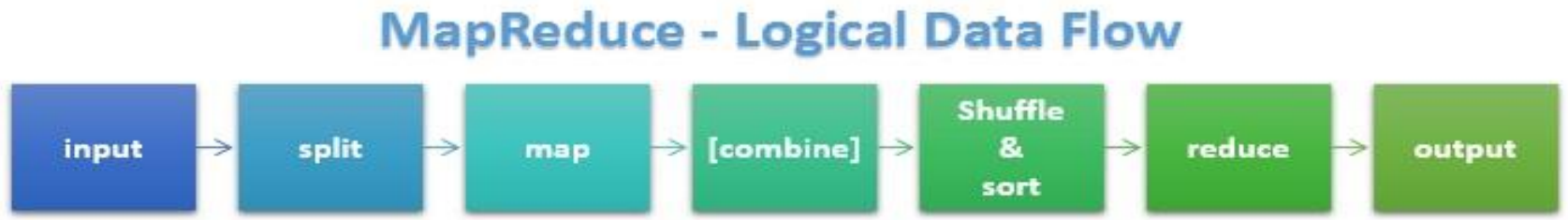
MapReduce

Software Framework for easily running applications	Processes large amount of data in parallel	Using large clusters having thousands of nodes	Nodes of commodity hardware	In a reliable and fault-tolerant manner

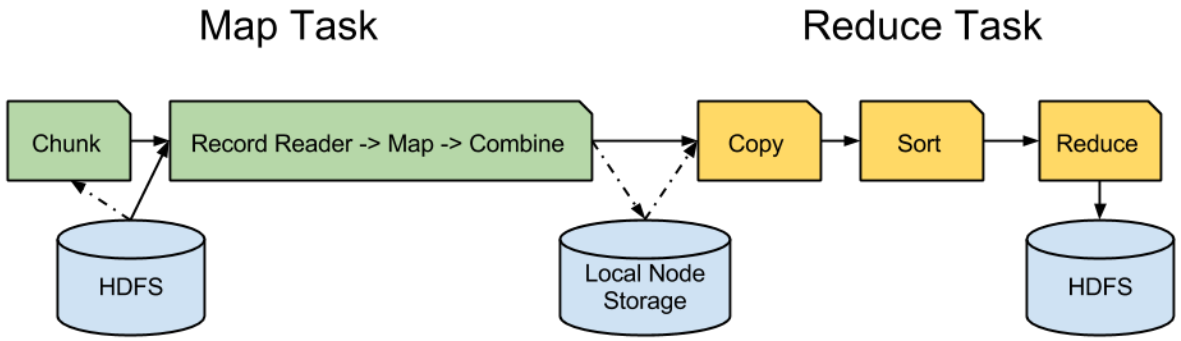


MapReduce Processing flow

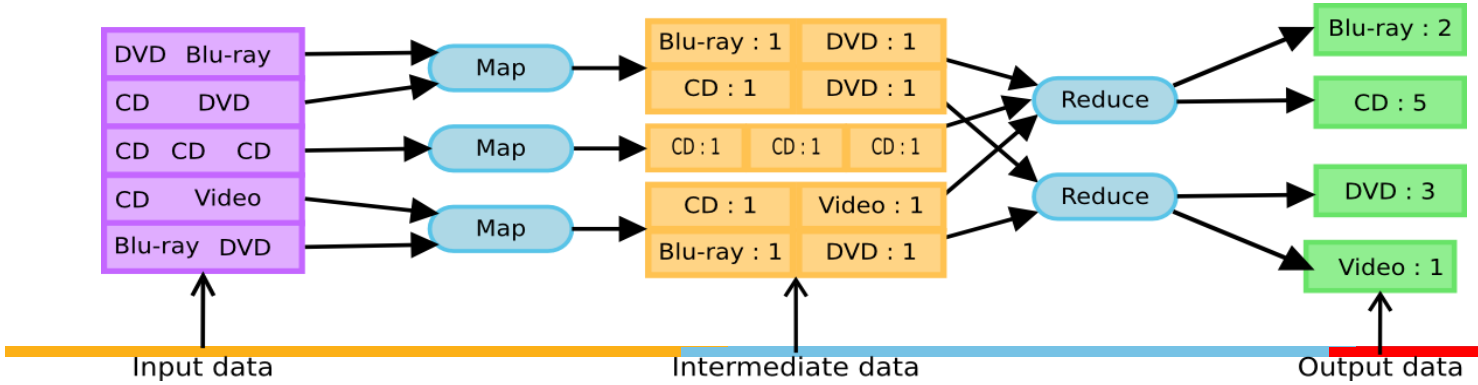
Logical flow:



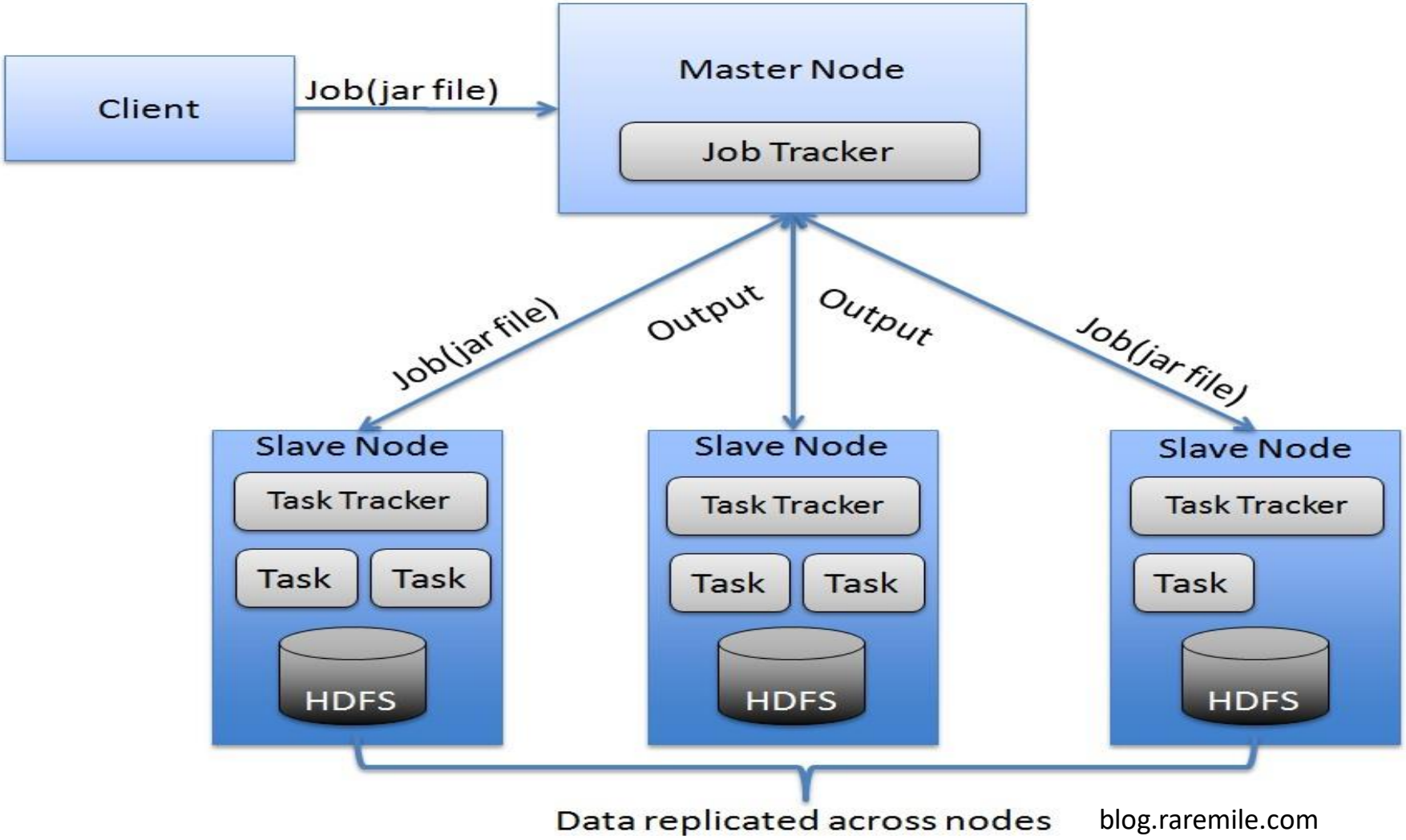
Two phases



Example:



Architecture Overview



Architecture (cont.)

NameNode/ Master Node:

- Stores metadata for the files, like the directory structure of a typical FS.
- The server holding the NameNode instance is quite crucial, as there is only one.
- Transaction log for file deletes/adds, etc.
- Handles creation of more replica blocks when necessary after a DataNode failure

DataNode/ Slave Node:

- Stores the actual data in HDFS
- Can run on any underlying filesystem (ext3/4, NTFS, etc)
- Notifies NameNode of what blocks it has
- NameNode replicates blocks 2x in local rack, 1x elsewhere

Architecture (cont.)

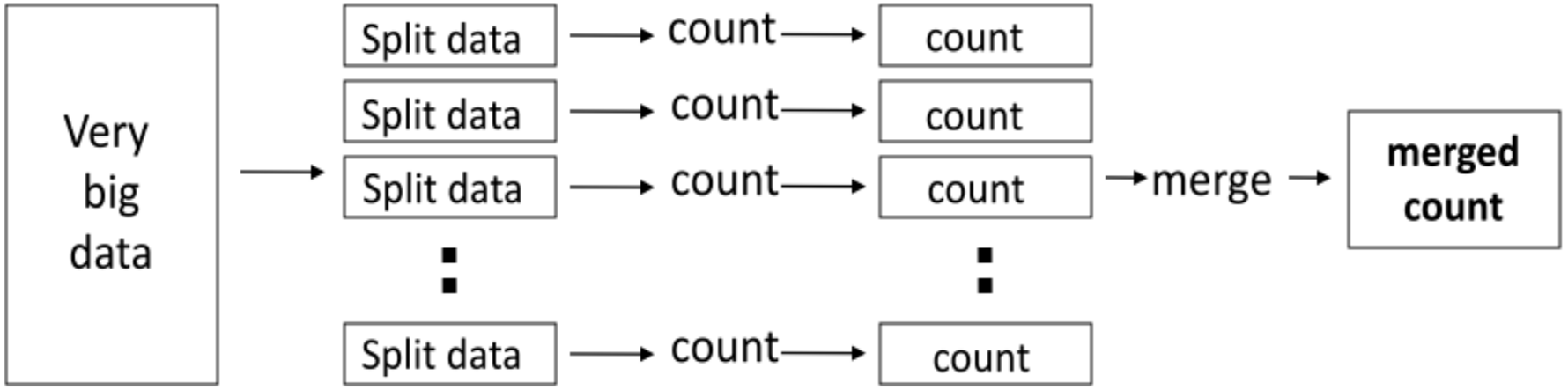
The Job Tracker:

- Central authority for the complete MapReduce cluster and responsible for scheduling and monitoring MapReduce jobs
- Responds to client request for job submission and status

The Task Tracker:

- Workers that accepts map and reduce tasks from job tracker, launches them and keeps track of their progress, reports the same to job tracker.
- Keeps track of resource usage of tasks and kills the tasks that overshoots their memory limits

Distributed Word Count



MapReduce Programming Model

Data type: key-value *records*

Map function:

$(K_{in}, V_{in}) \rightarrow list(K_{inter}, V_{inter})$

Reduce function:

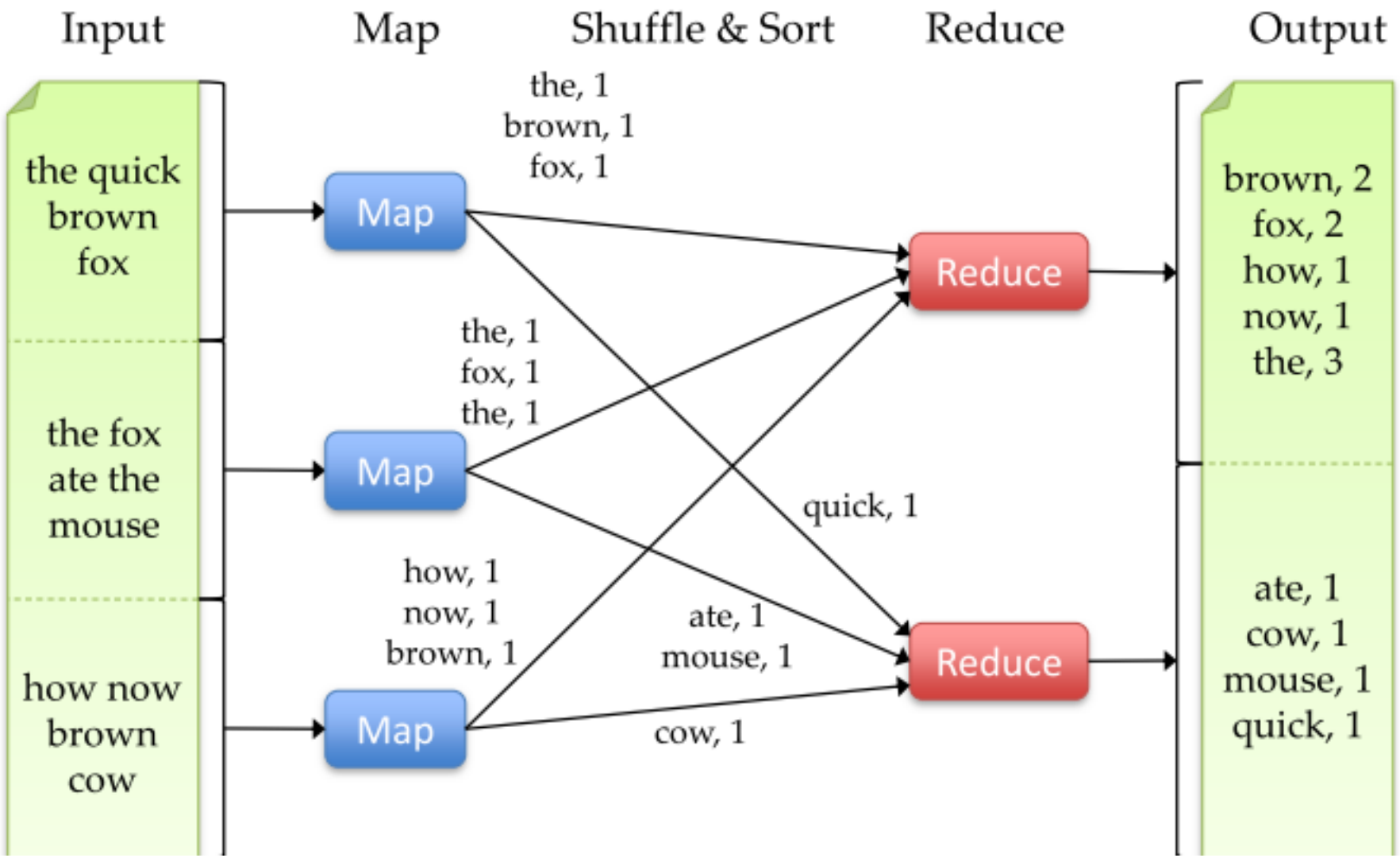
$(K_{inter}, list(V_{inter})) \rightarrow list(K_{out}, V_{out})$

Example: Word Count

```
def mapper(line):
    foreach word in line.split():
        output(word, 1)


def reducer(key, values):
    output(key, sum(values))
```

Word Count Execution



An Optimization: The Combiner

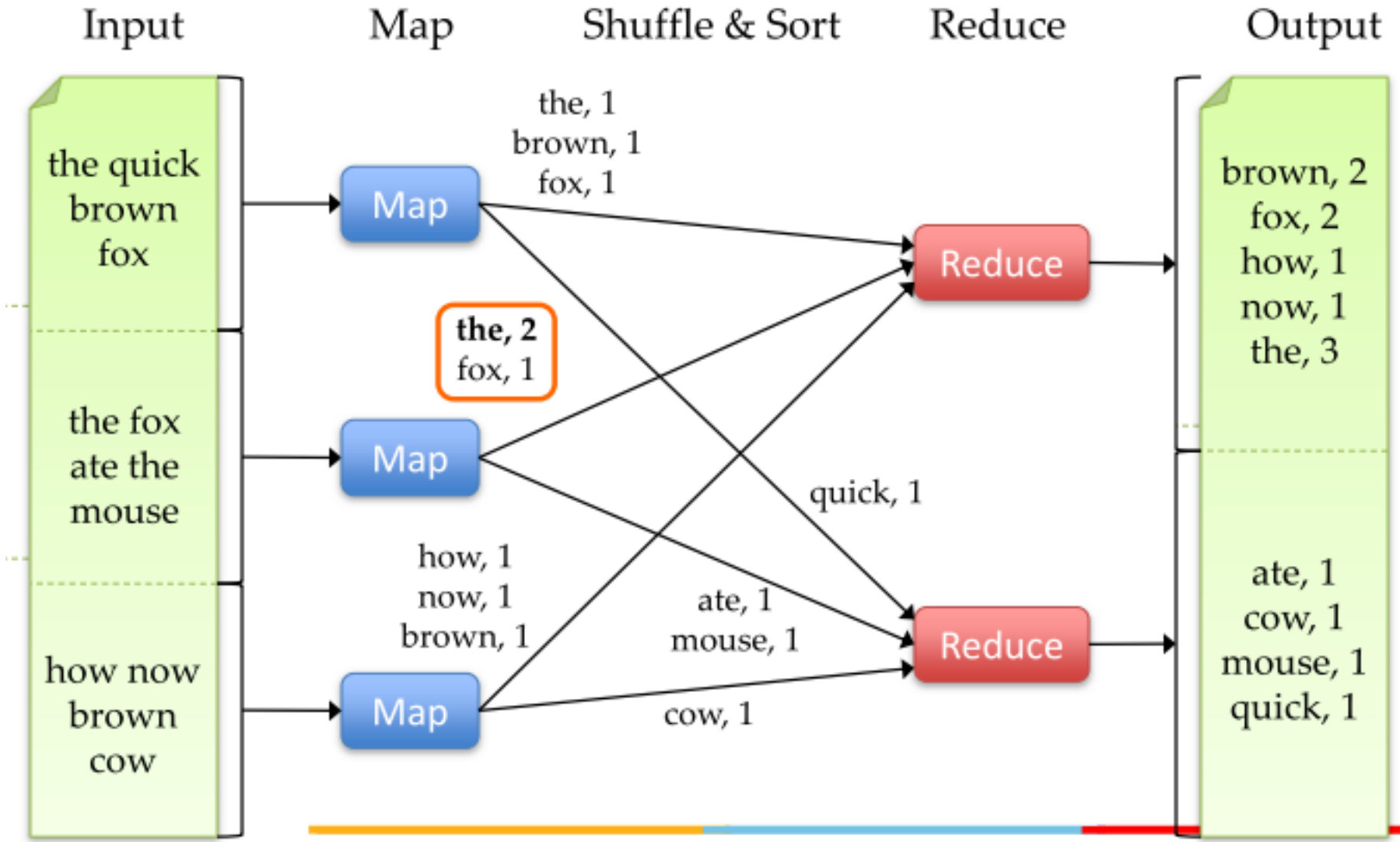
- Local reduce function for repeated keys produced by same map
- Decreases amount of intermediate data

• Example: local counting for Word Count:

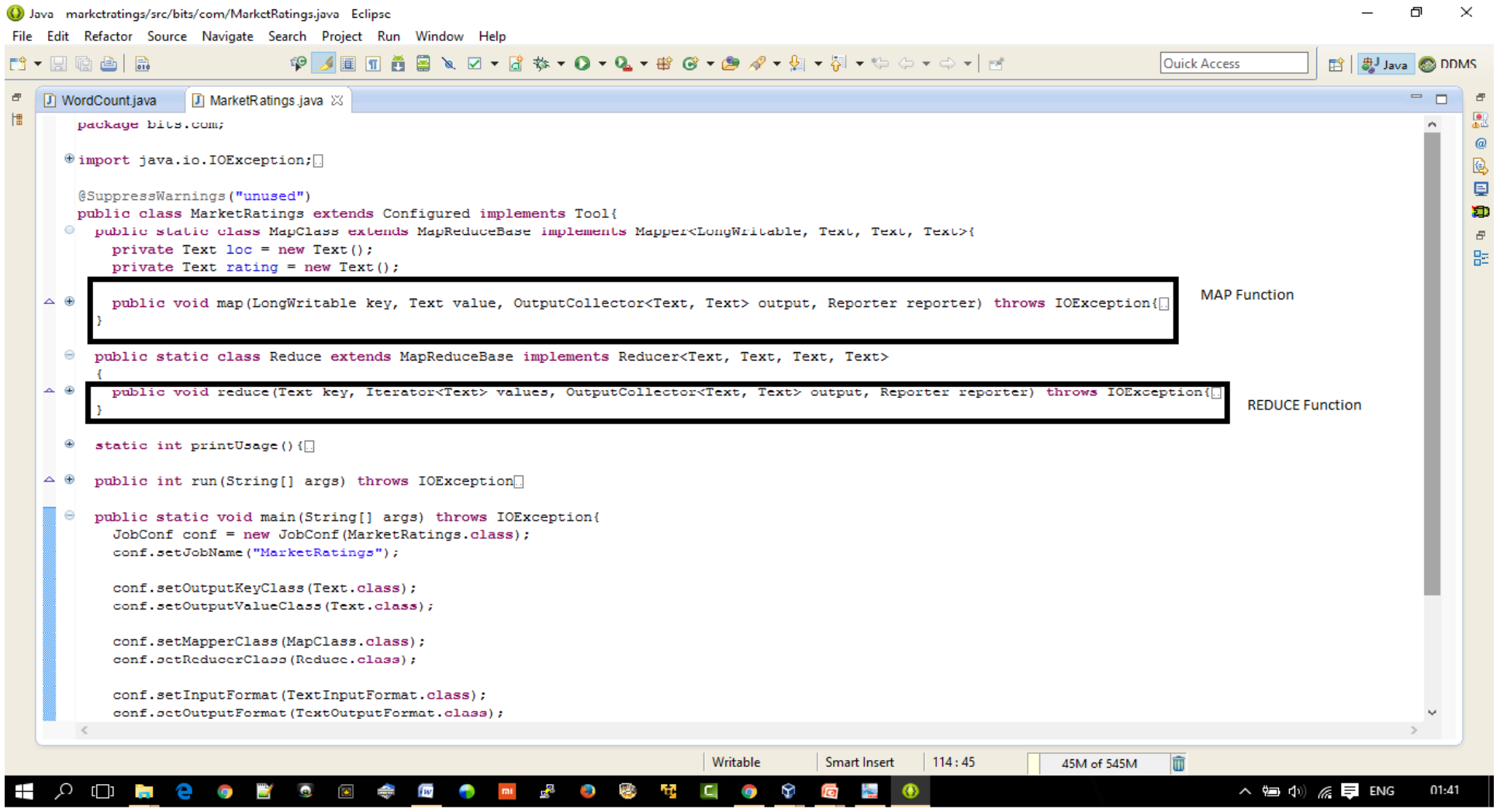
An Optimization: The Combiner

```
def combiner(key, values):
output(key, sum(values))
```

Word Count with Combiner



Snapshot of MarketRatings example and Program demo



MapReduce Execution Details

Mappers preferentially scheduled on same node or same rack as their input block

- Minimize network use to improve performance

Mappers save outputs to local disk before serving to reducers

- Allows recovery if a reducer crashes
- Allows running more reducers than # of nodes

Fault Tolerance in MapReduce

1. If a task crashes:
   – Retry on another node
      • OK for a map because it had no dependencies
      • OK for reduce because map outputs are on disk
   – If the same task repeatedly fails, fail the job or ignore that input block
2. If a node crashes:
   – Relaunch its current tasks on other nodes
   – Relaunch any maps the node previously ran
      • Necessary because their output files were lost along with the crashed node

Fault Tolerance in MapReduce

3. If a task is going slowly (straggler):
   – Launch second copy of task on another node
   – Take the output of whichever copy finishes first, and kill the other one
      • Critical for performance in large clusters

Challenges of Cloud Environment

Cheap nodes fail, especially when you have many
– Mean time between failures for 1 node = 3 years
– MTBF for 1000 nodes = 1 day
– Solution: Build fault tolerance into system

Commodity network = low bandwidth
– Solution: Push computation to the data

Programming distributed systems is hard
– Solution: Restricted programming model: users write data-parallel “map” and “reduce” functions and system handles work distribution and failures

Hadoo

Thanks Hadoop