Unit-III

Syllabus Content: Cloud Computing Mechanisms: Cloud Infrastructure, Logical Network Perimeter, Virtual Server, Cloud Storage Device, Cloud Usage Monitor, Specialized Cloud Mechanisms, Load Balancer, SLA Monitor, Failover System, Hypervisor, Automated Scaling Cloud Management Mechanisms, Resource Management System, SLA Management System, CASE STUDY examples.

From Book:-Thomas, Erl, Mahmood Zaigham, and Puttini Ricardo. "Cloud Computing Concepts, Technology & Architecture." (2013).

Logical Network Perimeter

- Cloud infrastructure mechanisms are foundational building blocks of cloud environments that establish primary artifacts to form the basis of fundamental cloud technology architecture.

- Defined as the isolation of a network environment from the rest of a communications network, the logical network perimeter establishes a virtual network boundary that can encompass and isolate a group of related cloud-based IT resources that may be physically distributed.

This mechanism can be implemented to:

Isolate IT resources in a cloud from non-authorized users

Isolate IT resources in a cloud from non-users

Isolate IT resources in a cloud from cloud consumers

Control the bandwidth that is available to isolated IT resources

Logical network perimeters are typically established via network devices that

Supply and control the connectivity of a data center and are commonly deployed

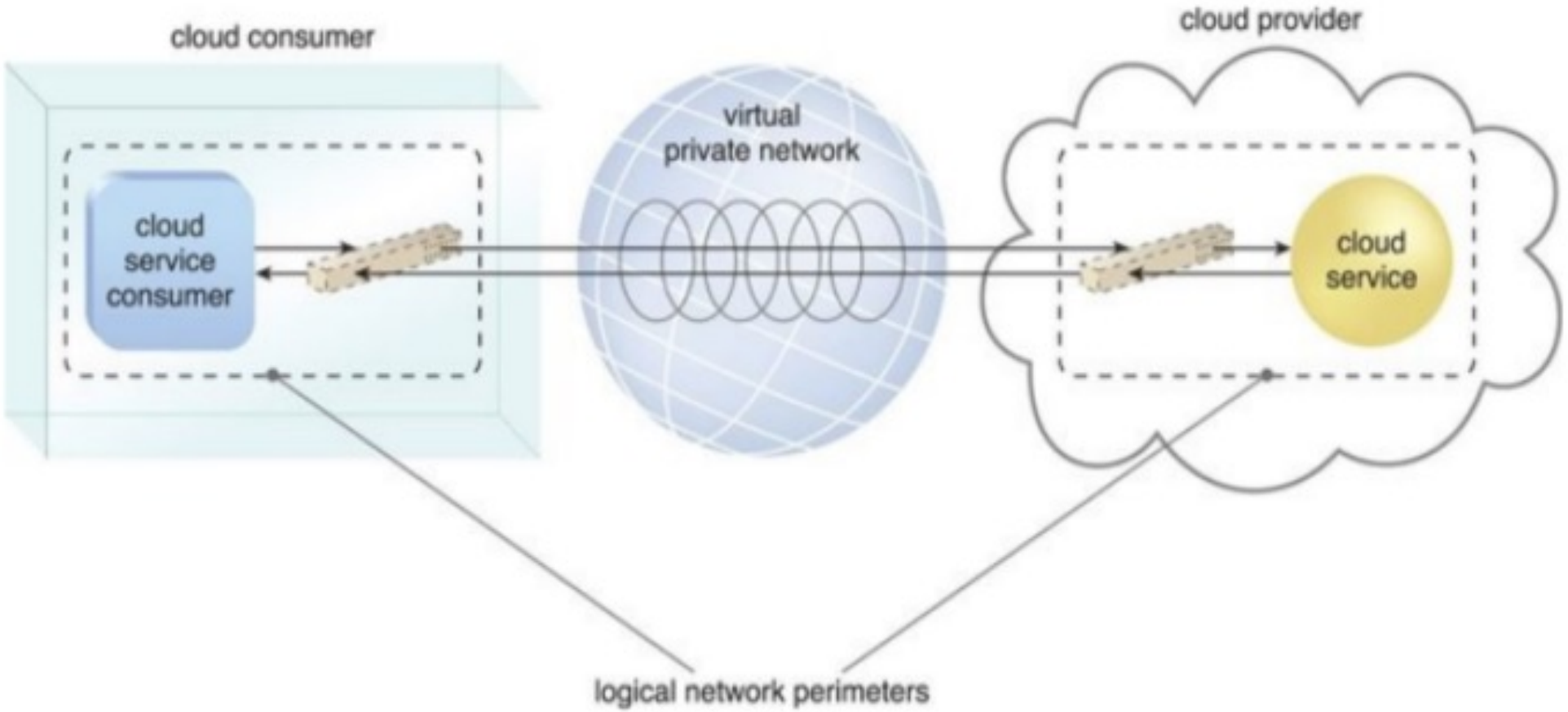
As virtualized IT environments that include:-

Virtual Firewall – An IT resource that actively filters network traffic to

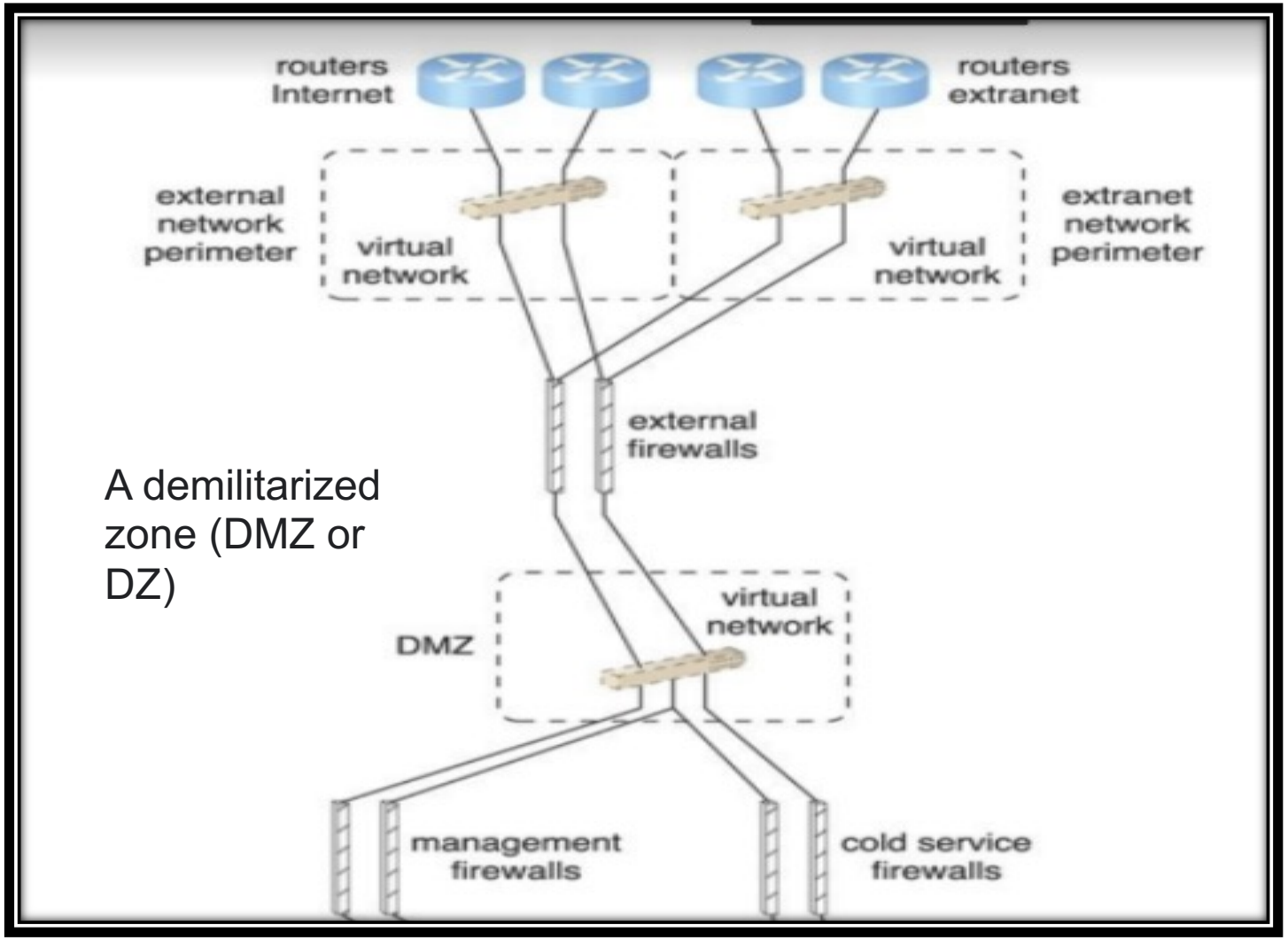
and from the isolated network while controlling its interactions with the Internet.

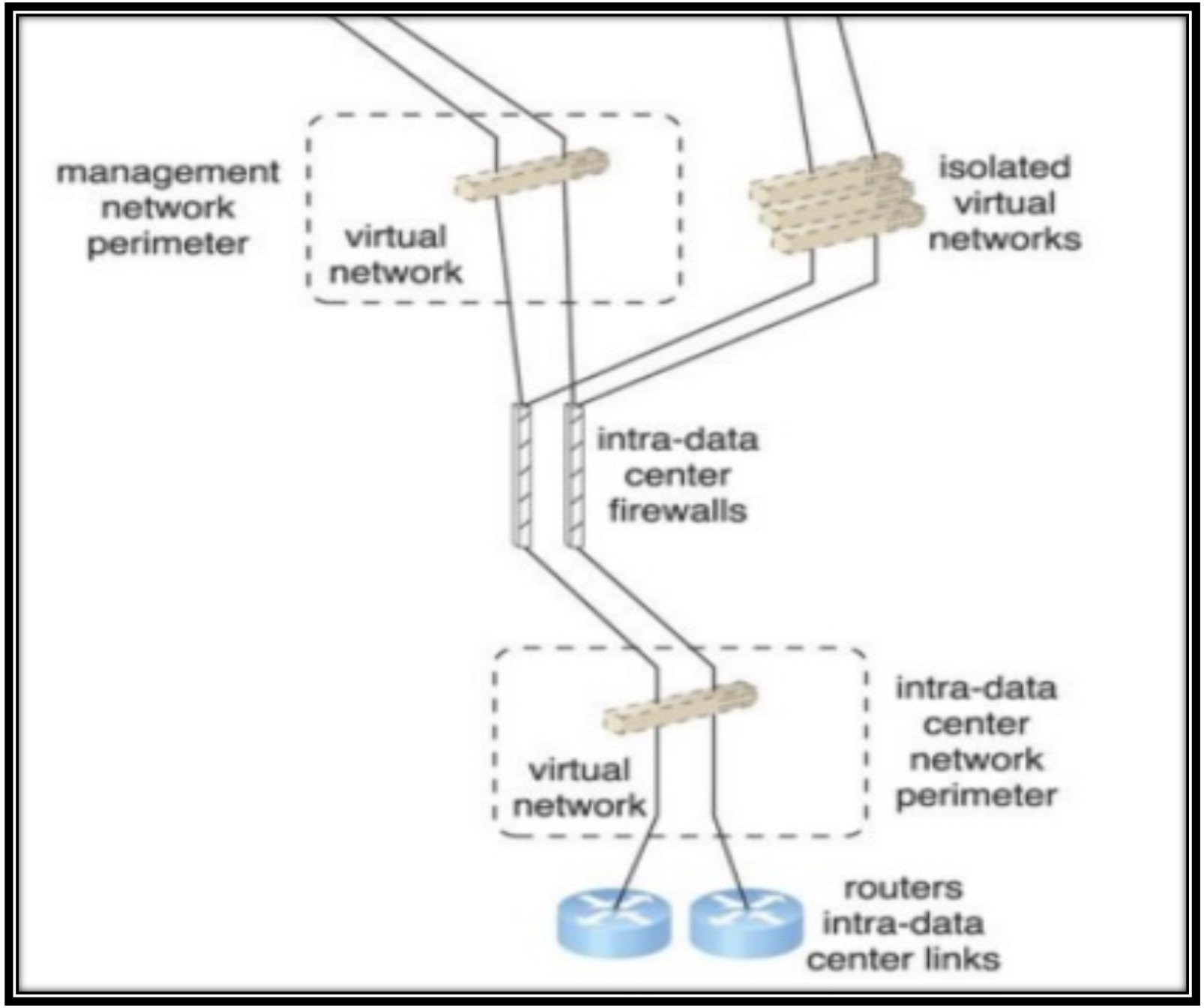
Virtual Network – Usually acquired through VLANs, this IT resource isolates the network environment within the data center infrastructure.

Placement of the VPN [177]



Another Case: A logical network layout is established through a set of logical
network perimeters using various firewalls and virtual networks.

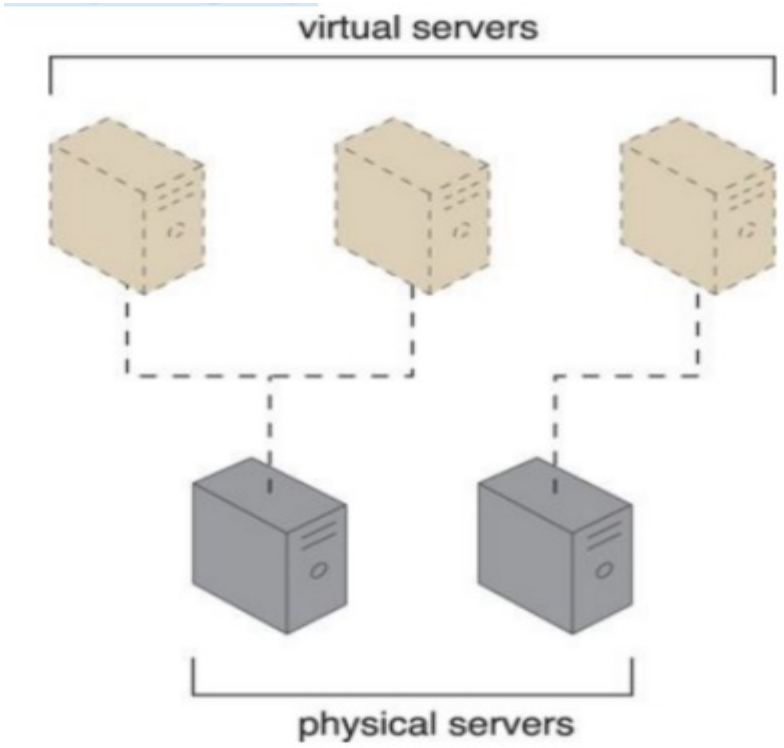



management
network
perimeter
virtual
network
isolated
virtual
networks
intra-data
center
firewalls
virtual
network
intra-data
center
network
perimeter
routers
intra-data
center links

Virtual Server

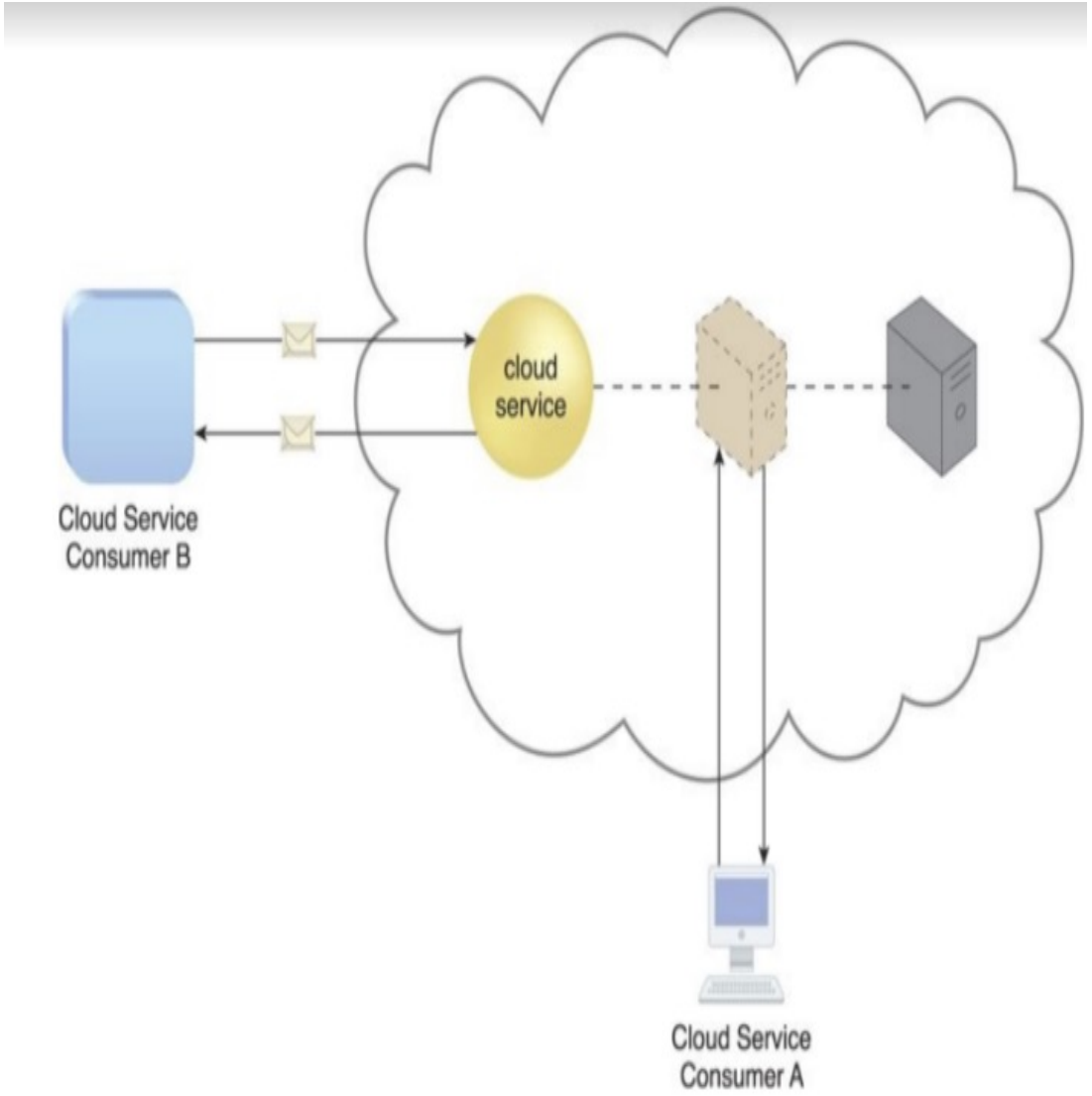
A virtual server is a form of virtualization software that emulates a physical server. Virtual servers are used by cloud providers to share the same physical server with multiple cloud consumers by providing cloud consumers with individual virtual server instances.

Shows three virtual servers being hosted by two physical servers. The number of instances a given physical server can share is limited by its capacity.

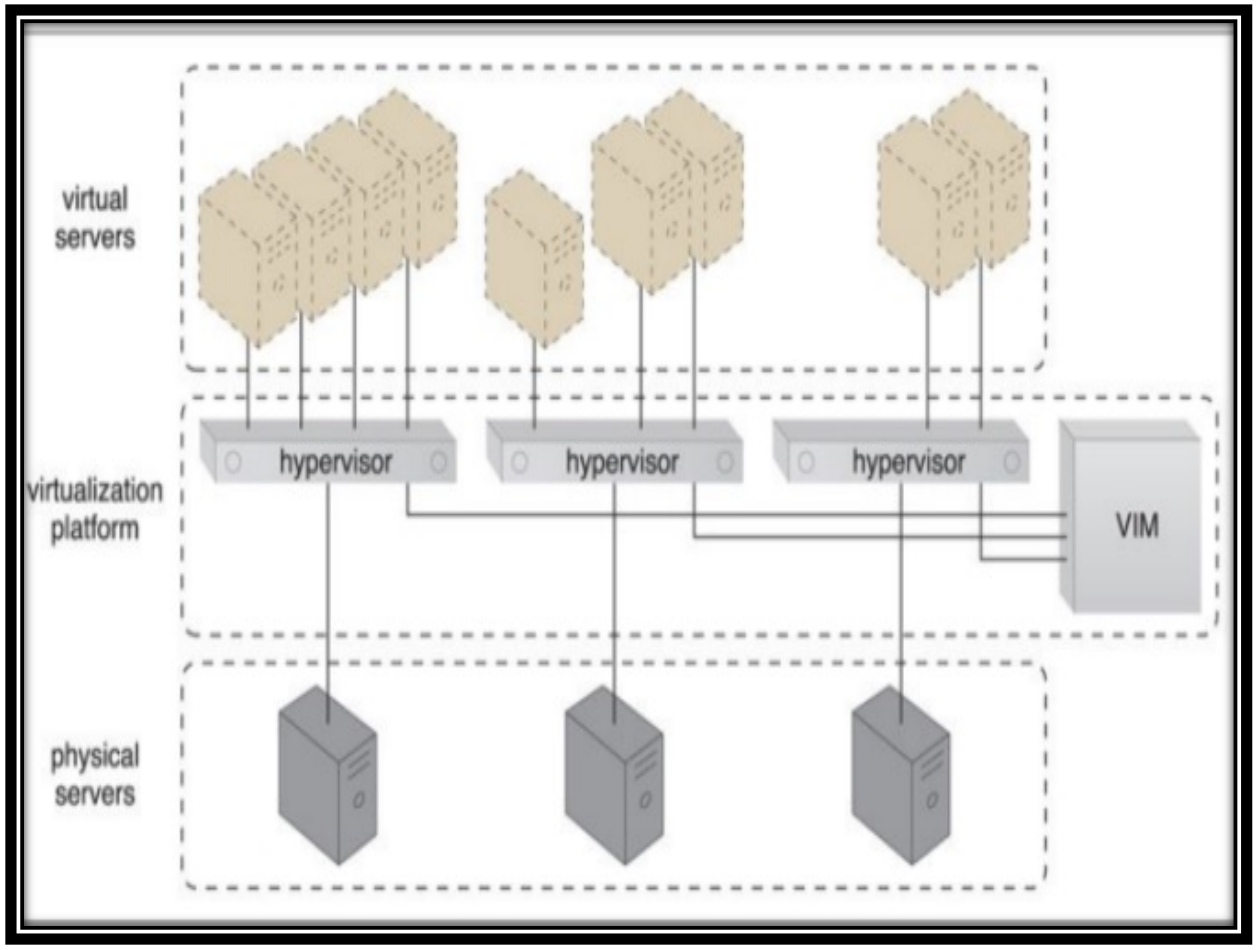


- Each virtual server can host numerous IT resources, cloud-based solutions, and various other cloud computing mechanisms. The instantiation of virtual servers from image files is a resource allocation process that can be completed rapidly and on-demand.

Cloud consumers that install or lease virtual servers can customize their environments independently from other cloud consumers that may be using virtual servers hosted by the same underlying physical server. Figure depicts a virtual server that hosts a cloud service being accessed by Cloud Service Consumer B, while Cloud Service Consumer A accesses the virtual server directly to perform an administration task.



VIM



How this will fulfill the requirements ?

- Small Virtual Server Instance – 1 virtual processor core, 4 GB of

virtual RAM, 20 GB of storage space in the root file system

• Medium Virtual Server Instance – 2 virtual processor cores, 8 GB of
virtual RAM, 20 GB of storage space in the root file system

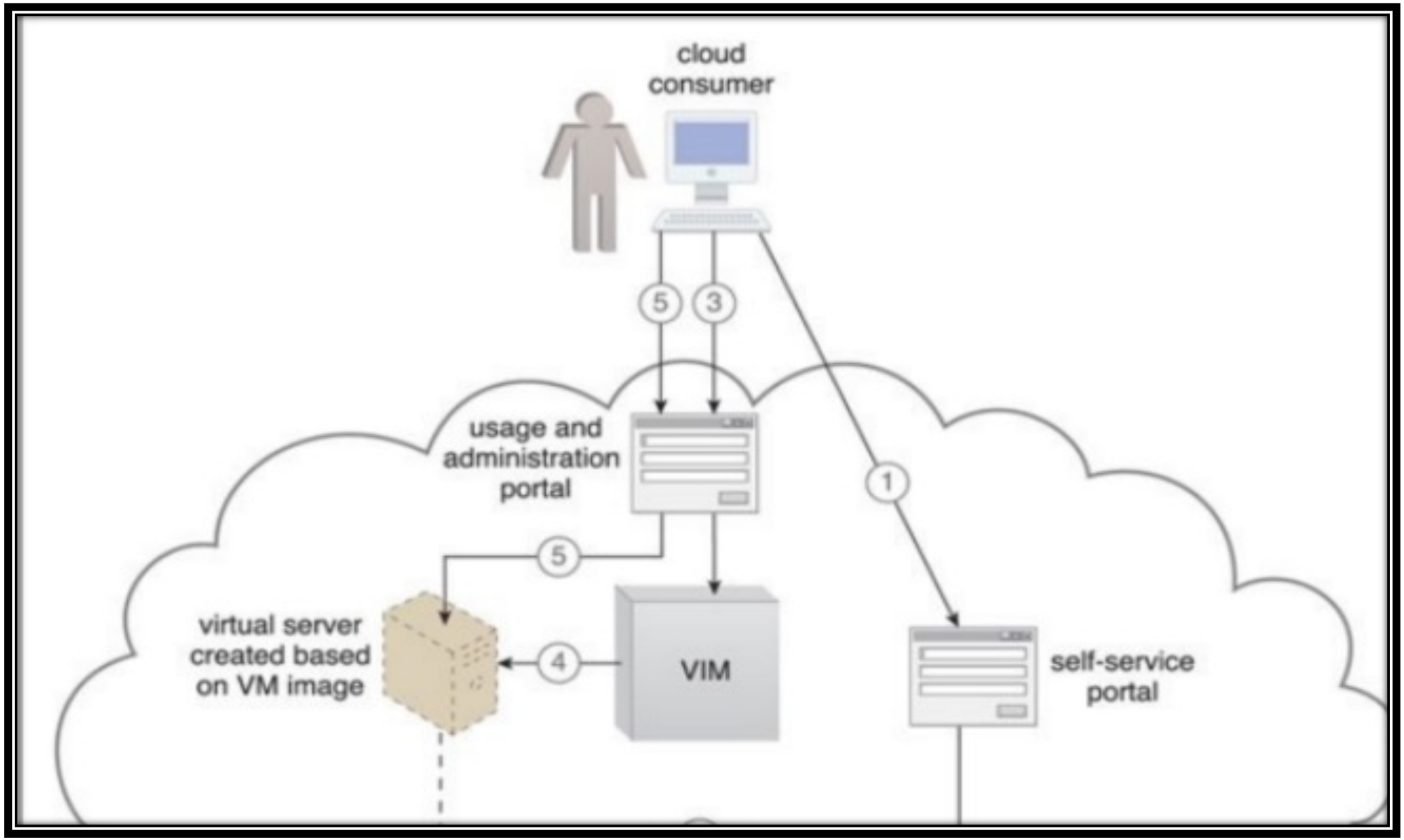
• Large Virtual Server Instance – 8 virtual processor cores, 16 GB of
virtual RAM, 20 GB of storage space in the root file system

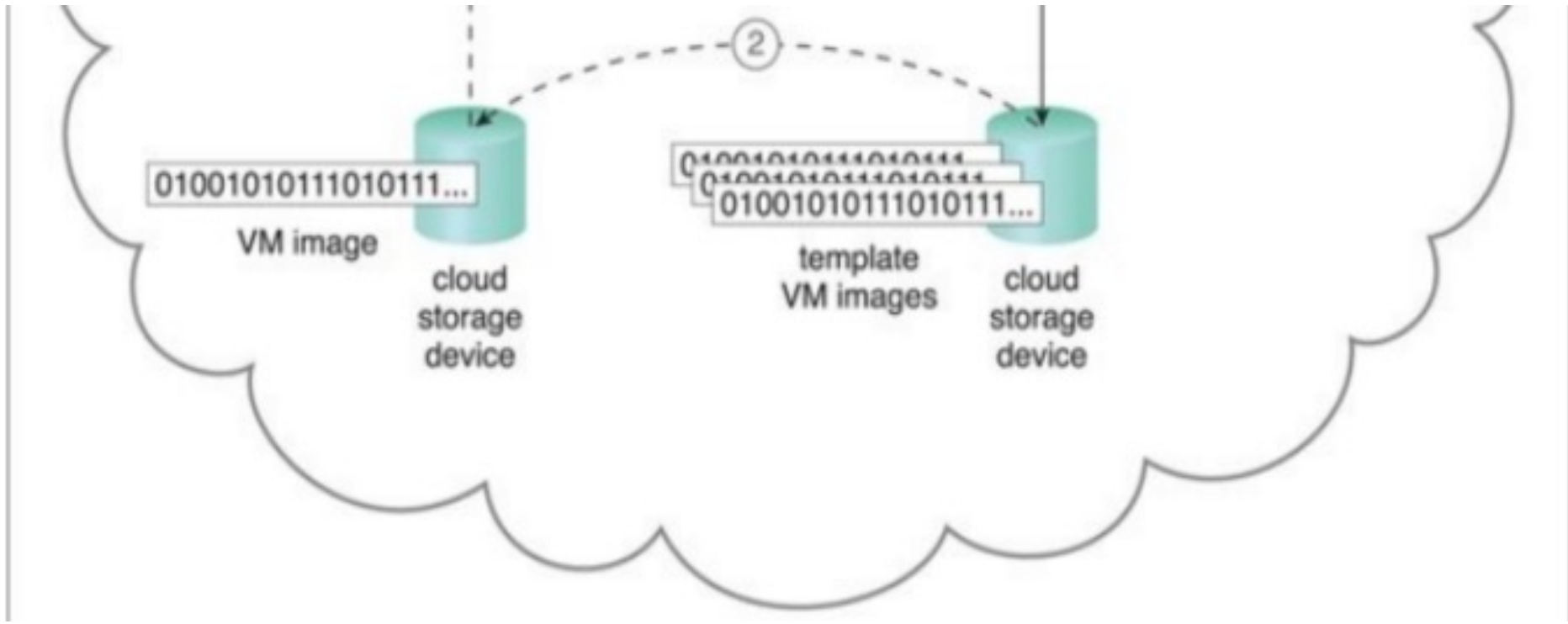
• Memory Large Virtual Server Instance – 8 virtual processor cores, 64
GB of virtual RAM, 20 GB of storage space in the root file system

Cloud Consumer and Cloud Infrastructure

The cloud consumer uses the self-service portal to select a template virtual server for creation (1). A copy of the corresponding VM image is created in a cloud consumer-controlled cloud storage device (2). The cloud consumer initiates the virtual server using the usage and administration portal (3), which interacts with the VIM to create the virtual server instance via the underlying hardware (4). The cloud consumer is able to use and customize the virtual server via other features on the usage and administration portal (5).

Its diagram-----→


cloud
consumer
5
3
usage and
administration
portal
1
5
virtual server
created based
on VM image
4
VIM
self-service
portal


2
01001010111010111...
VM image
cloud
storage
device
01001010111010111...
template
VM images
cloud
storage
device

Cloud Storage Device

The cloud storage device mechanism represents storage devices that are designed specifically for cloud-based provisioning. **Instances of these devices can be virtualized, similar to how physical servers can spawn virtual server images.** Cloud storage devices are commonly able to provide fixed-increment capacity allocation in support of the pay-per-use mechanism. Cloud storage devices can be exposed for remote access via cloud storage services.

A primary concern related to cloud storage is the security, integrity, and confidentiality of data, which becomes more prone to being compromised when entrusted to external cloud providers and other third parties. There can also be legal and regulatory implications that result from relocating data across geographical or national boundaries. Another issue applies specifically to the performance of large databases. LANs provide locally stored data with network reliability and latency levels that are superior to those of WANs.

Cloud Storage Levels

- Cloud storage device mechanisms provide common logical units of data storage, such as:

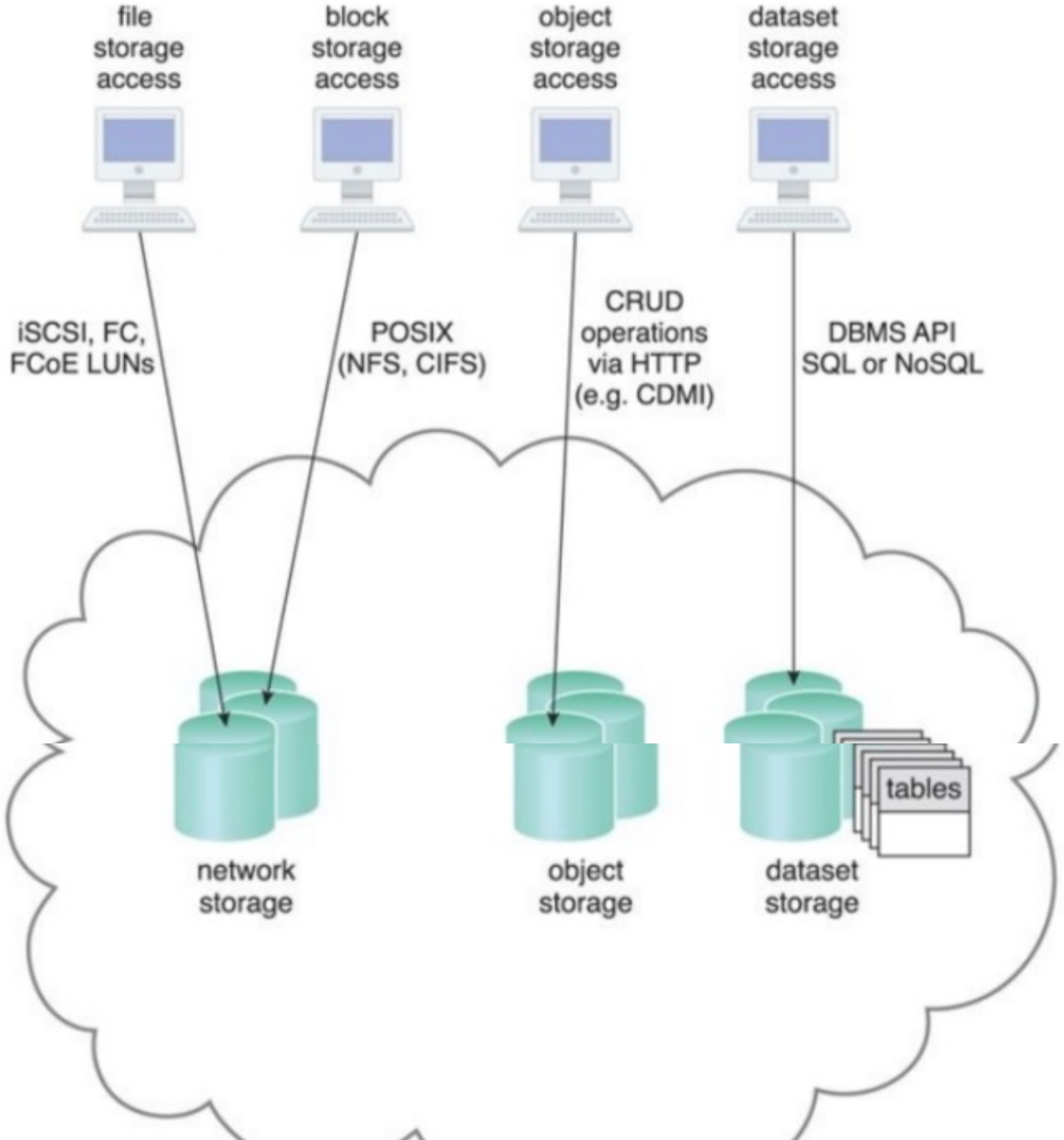
Files – Collections of data are grouped into files that are located in folders.

Blocks – The lowest level of storage and the closest to the hardware, a block is the smallest unit of data that is still individually accessible.

Datasets – Sets of data are organized into a table-based, delimited, or record format.

Objects – Data and its associated metadata are organized as Web-based resources.

Each of these data storage levels is commonly associated with a certain type of technical interface which corresponds to a particular type of cloud storage device and cloud storage service used to expose its API.



Different storage types and different APIs

Network Storage Interfaces:-Legacy network storage most commonly falls under the category of network storage interfaces. It includes storage devices in compliance with industry standard protocols, such as SCSI (Small Computer System Interface) for storage blocks and the server message. block (SMB), common Internet file system (CIFS), and network file system(NFS) for file and network storage.

Object Storage Interfaces: Various types of data can be referenced and stored as Web resources. This is referred to as object storage, which is based on technologies that can support a range of **data and media types.** *Cloud Storage Device mechanisms that implement this interface can typically be accessed via REST or Web service- based cloud services using HTTP as the prime protocol. The Storage Networking Industry Association's Cloud Data Management Interface (SNIA's CDMI)*

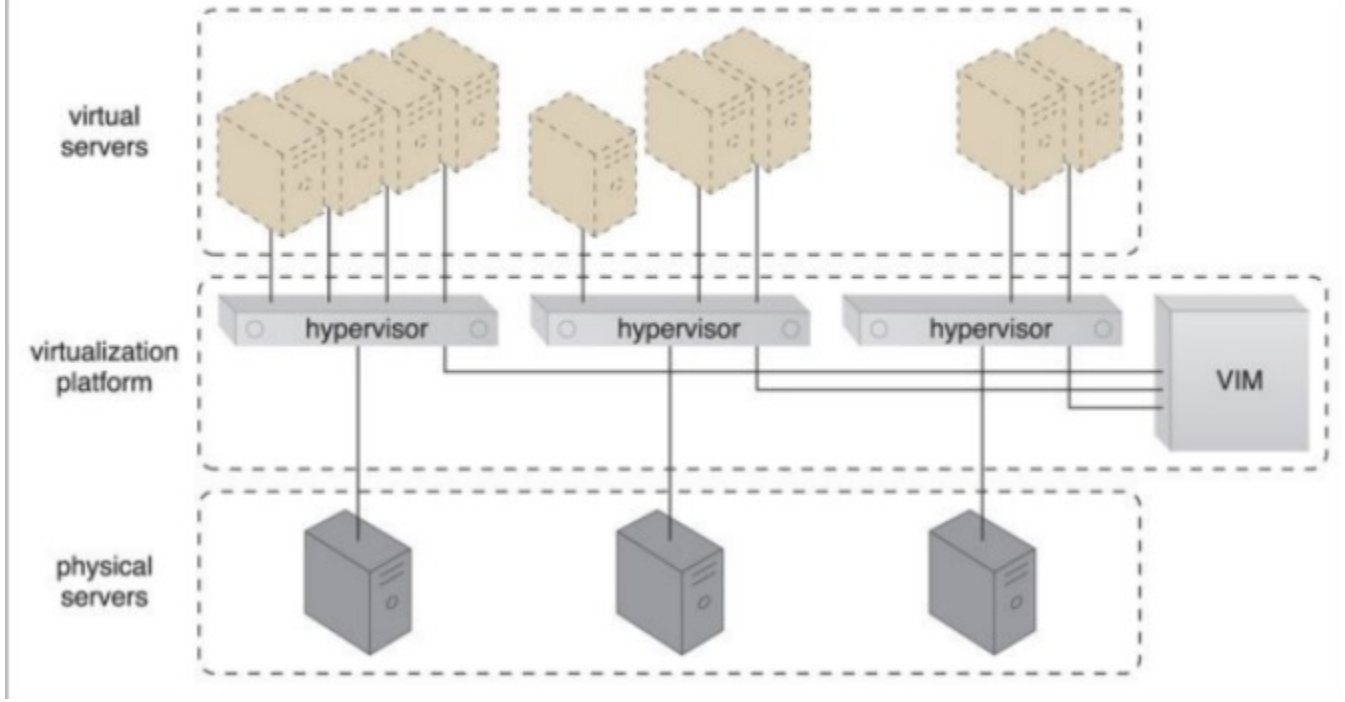
supports the use of object storage interfaces.

Database Storage Interfaces: Traditionally, many on-premise IT environments store data using relational databases or relational database management systems (RDBMSs).

Non-relational storage (also commonly referred to as NoSQL storage) moves away from the traditional relational database model in that it establishes a“loose” structure for stored data with less emphasis on defining relationships and realizing data normalization.

Work of the VIM

Figure 7.7 depicts several virtual servers running over physical servers, all of which are jointly controlled by a central VIM.

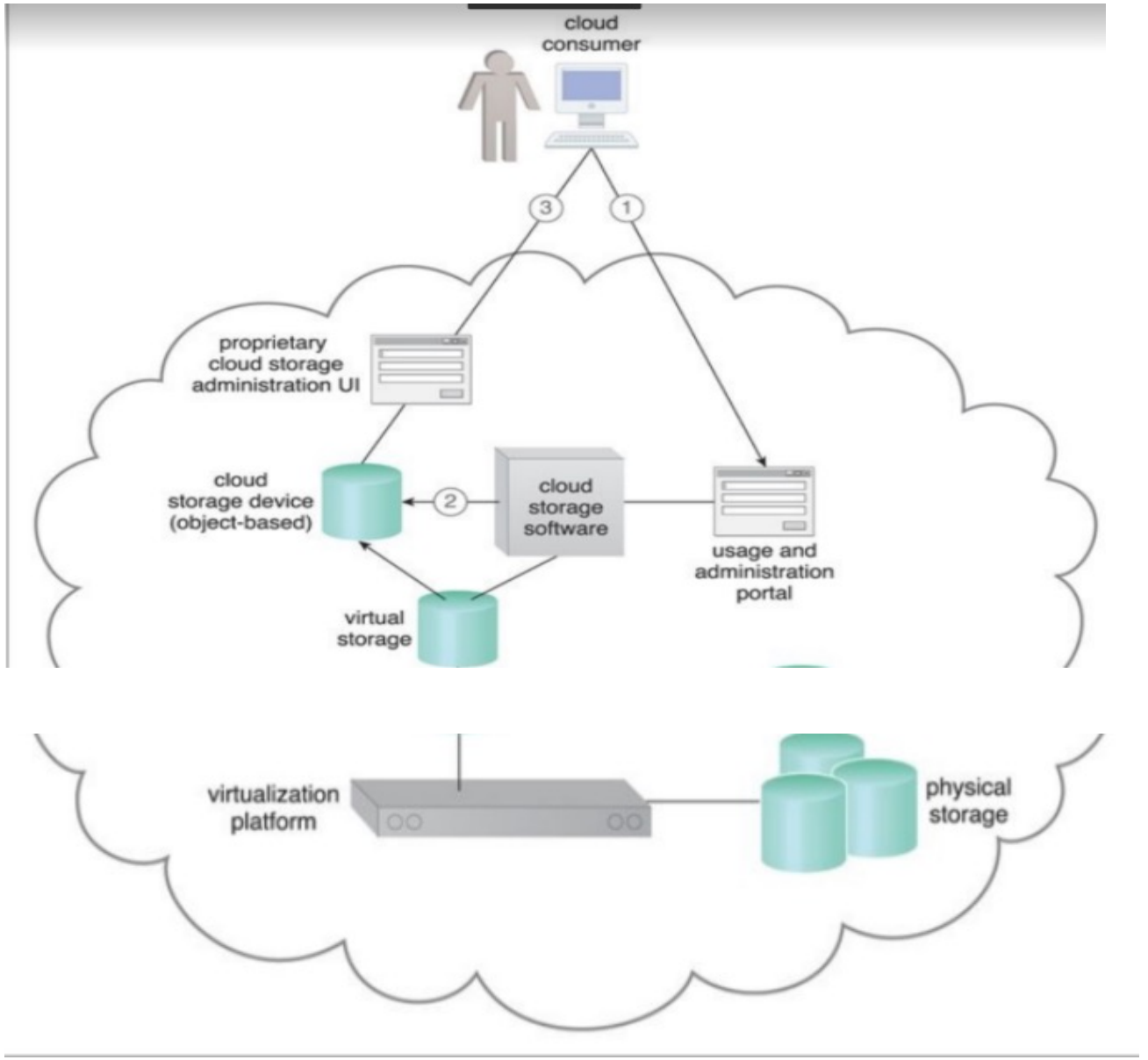


The VIM carries out several tasks:

- Allocating resources in accordance with traffic engineering rules.
- Support for defining **operational rules-**-IMP.
- Definition of hub-to-facility mapping.
- Providing information for provisioning virtual infrastructure orchestration (VIO).

The object-based cloud storage device has an underlying storage system with *variable storage capacity*, which is directly controlled by a software component that also exposes the interface.

This software enables the creation of i*solated cloud storage devices* that are allocated to cloud consumers. The storage system uses a security credential management system to administer user-based access control to the device's data objects



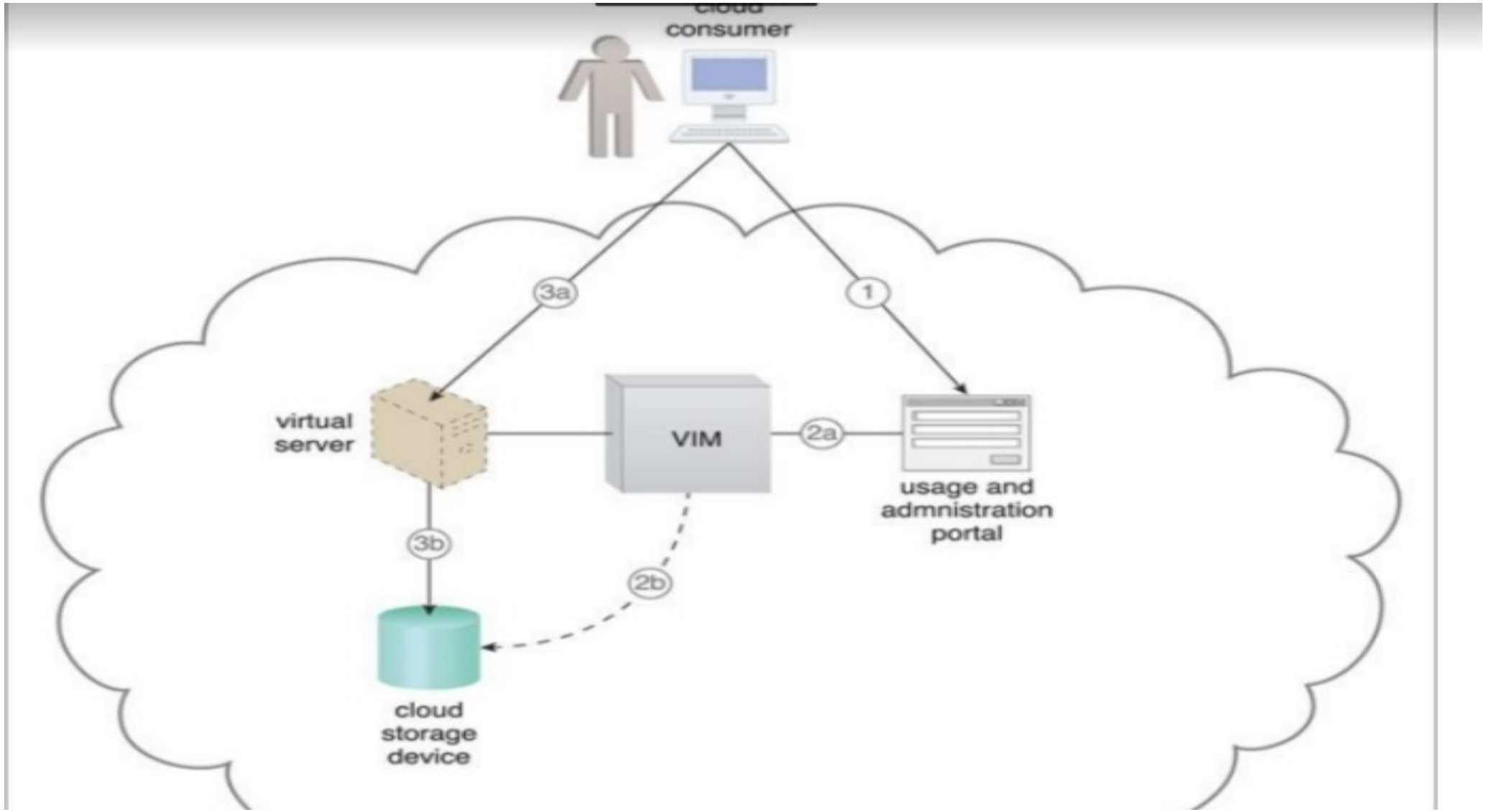
The cloud consumer interacts with the usage and administration portal to create a cloud storage device and define access control policies (1). The usage and administration portal ***interact with the cloud storage software to create the cloud storage device instance and apply the required access policy*** to its data objects (2). Each data object is assigned to a ***cloud storage device and all of the data objects are stored in the same virtual storage volume***. The cloud consumer uses the proprietary **cloud storage device UI to interact directly** with the data objects (3).

Cloud consumer and administration portal

The cloud consumer uses the usage and administration portal to create and assign a cloud storage device to an existing virtual server (1***). The usage and administration portal interacts with the VIM software*** (2a), which ***creates and configures the appropriate LUN***

[logical unit number (**LUN**) is a slice or portion of a configured set of disks ***]*** (2b).

Each cloud storage device uses a separate ***LUN controlled by the virtualization platform***. The cloud consumer remotely ***logs into the virtual server directly (3a) to access the cloud storage device*** (3b).


consumer
3a
1
virtual
server
VIM
2a
usage and
admnistration
portal
3b
2b
cloud
storage
device

Cloud Usage Monitor

The cloud usage monitor mechanism is a ***lightweight and autonomous software program responsible for collecting and processing IT resource usage data.***

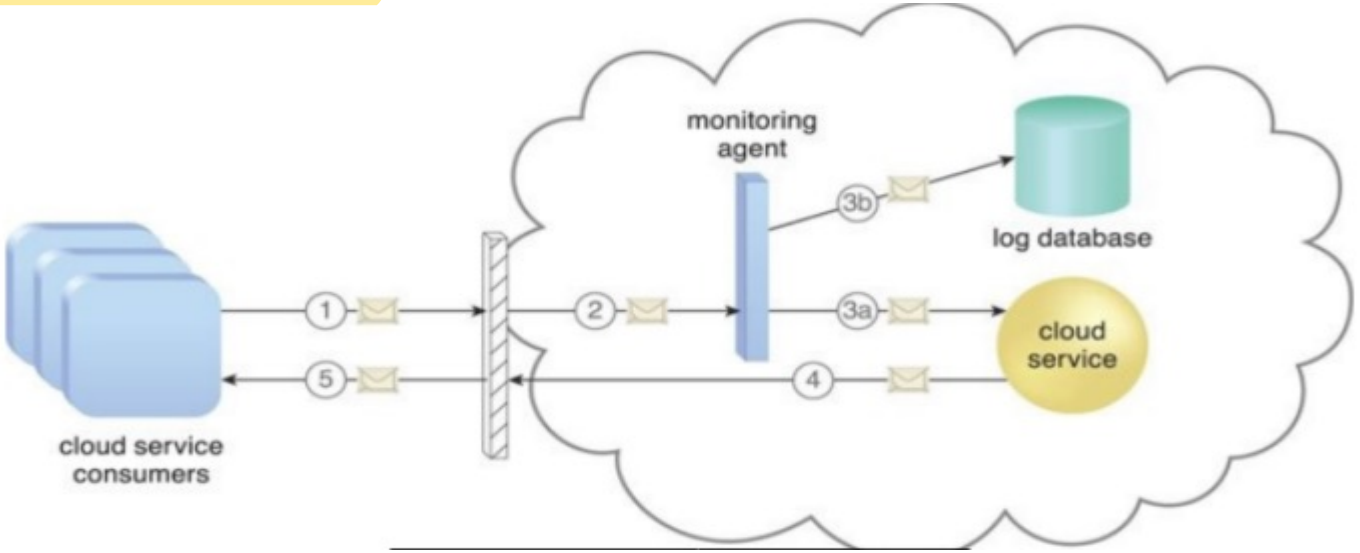
Depending on the type of usage metrics they are designed to collect and the manner in which usage data needs to be ***collected, cloud usage monitors can exist in different formats.***

The upcoming sections describe three common agent-based implementation formats.

Each can be designed to forward collected usage data to a log database for post-processing and reporting purposes.

Monitoring agent

- A monitoring agent is an ***intermediary, event-driven program*** that exists as a service agent and resides along existing communication paths to transparently monitor and analyze dataflows.
- This type of cloud usage monitor is commonly used to measure network traffic and message metrics.



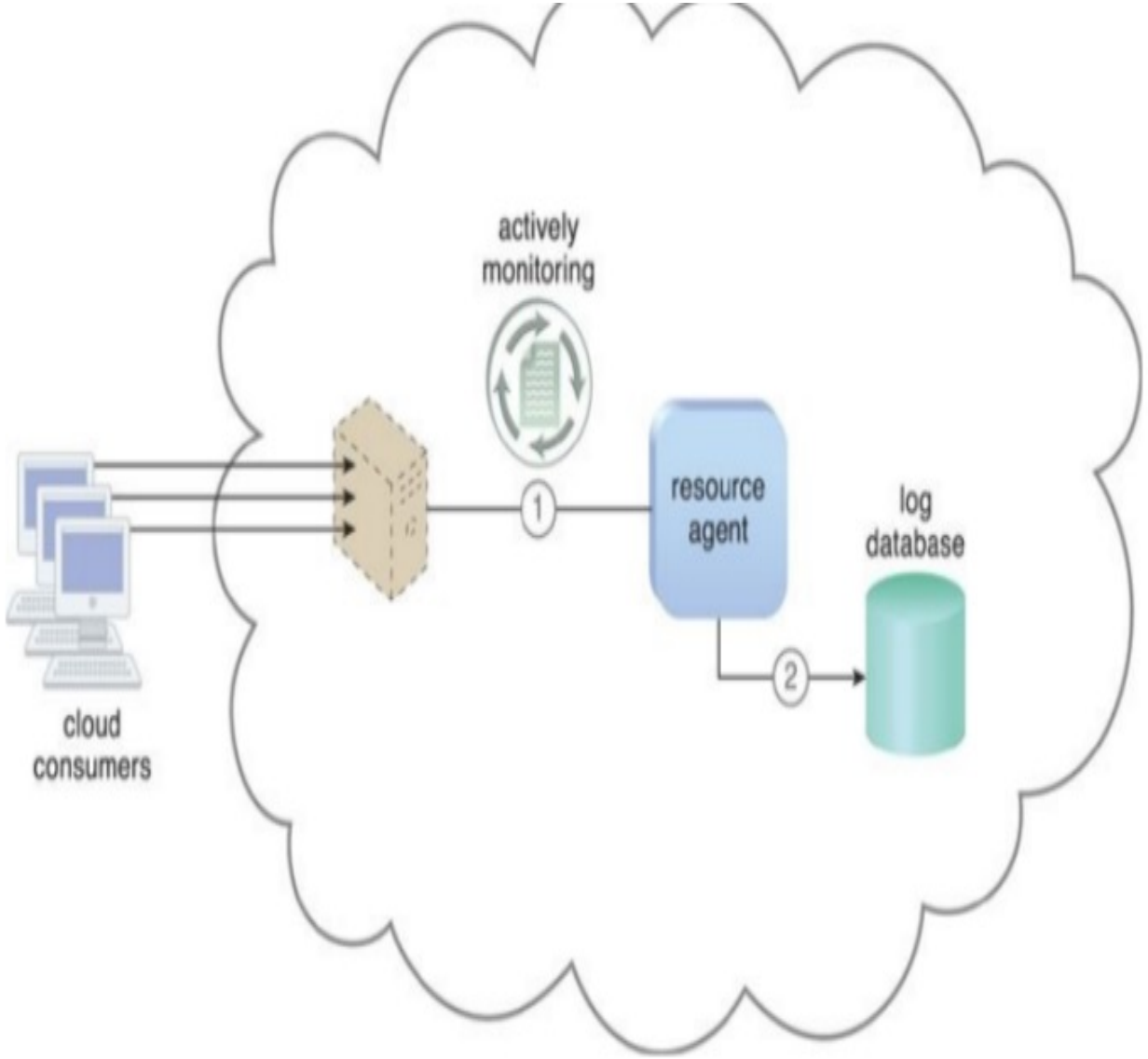
- A cloud service consumer sends a request message to a cloud service (1).
- The monitoring agent intercepts the message to collect relevant usage data (2) before allowing it to continue to the cloud service (3a).
- The monitoring agent stores the collected usage data in a log database (3b).
- The cloud service replies with a response message (4) that is sent back to the cloud.

Resource Agent

A resource agent is a processing module that collects usage data by having event-driven interactions with specialized resource software.

This module is used to monitor usage metrics based on pre-defined, observable events at the resource software level, such as initiating, suspending, resuming, and ***vertical scaling***.

Meaning of horizontal and vertical scaling:-**Horizontal scaling** means **scaling** by adding more machines to your pool of resources (also described as "**scaling** out"), whereas **vertical scaling** refers to **scaling** by adding more power (e.g. CPU, RAM) to an existing machine (also described as "**scaling** up")

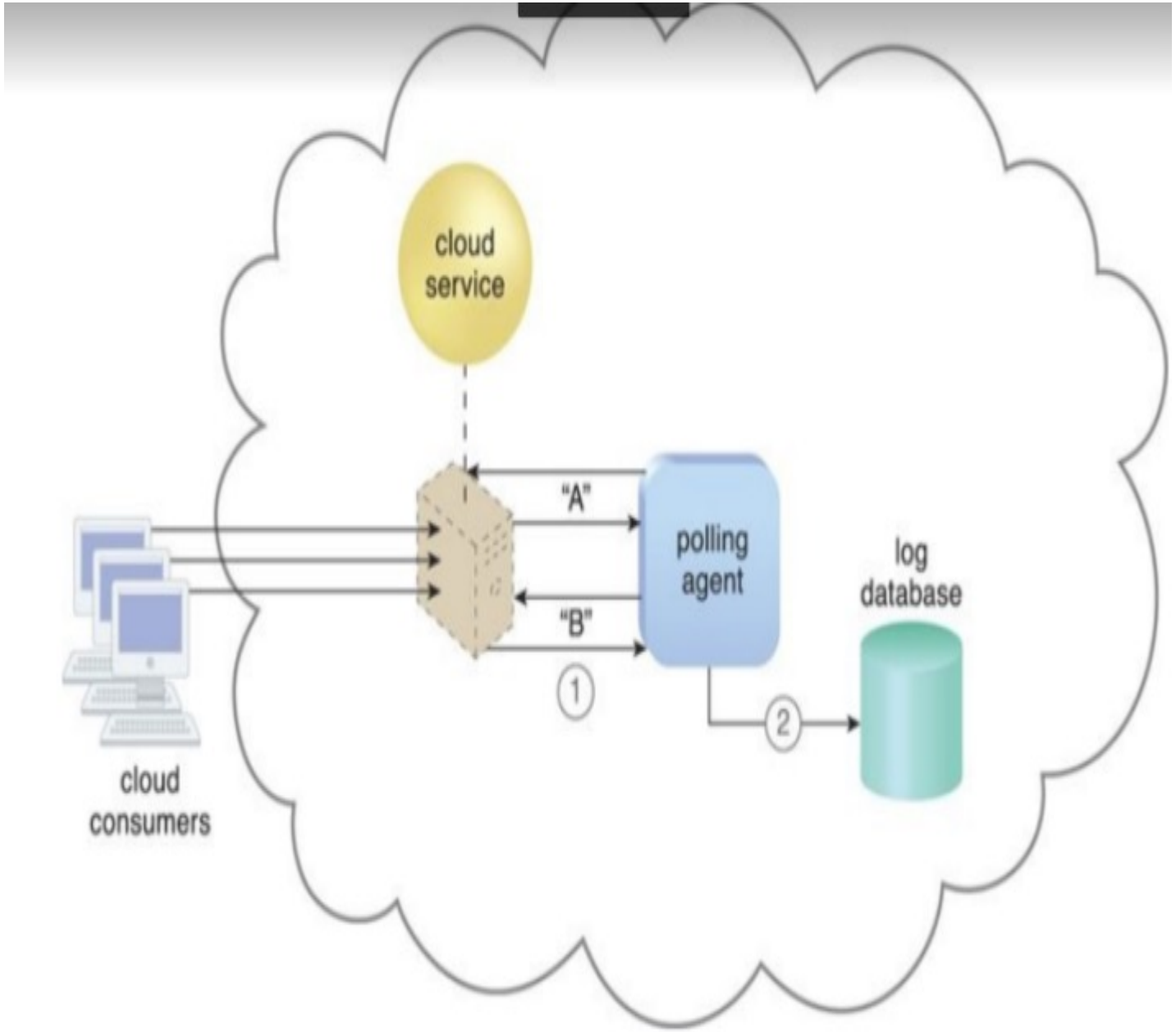


The resource agent is actively monitoring a virtual server and detects an increase in usage (1). The resource agent receives a notification from the underlying resource management program that the virtual server is being scaled up and stores the collected usage data in a log database, as per its monitoring metrics (2).

Polling Agent

A polling agent is a processing module that collects cloud service usage data by polling IT resources.

This type of cloud service monitor is commonly used to periodically monitor IT resource status, such as uptime and downtime.

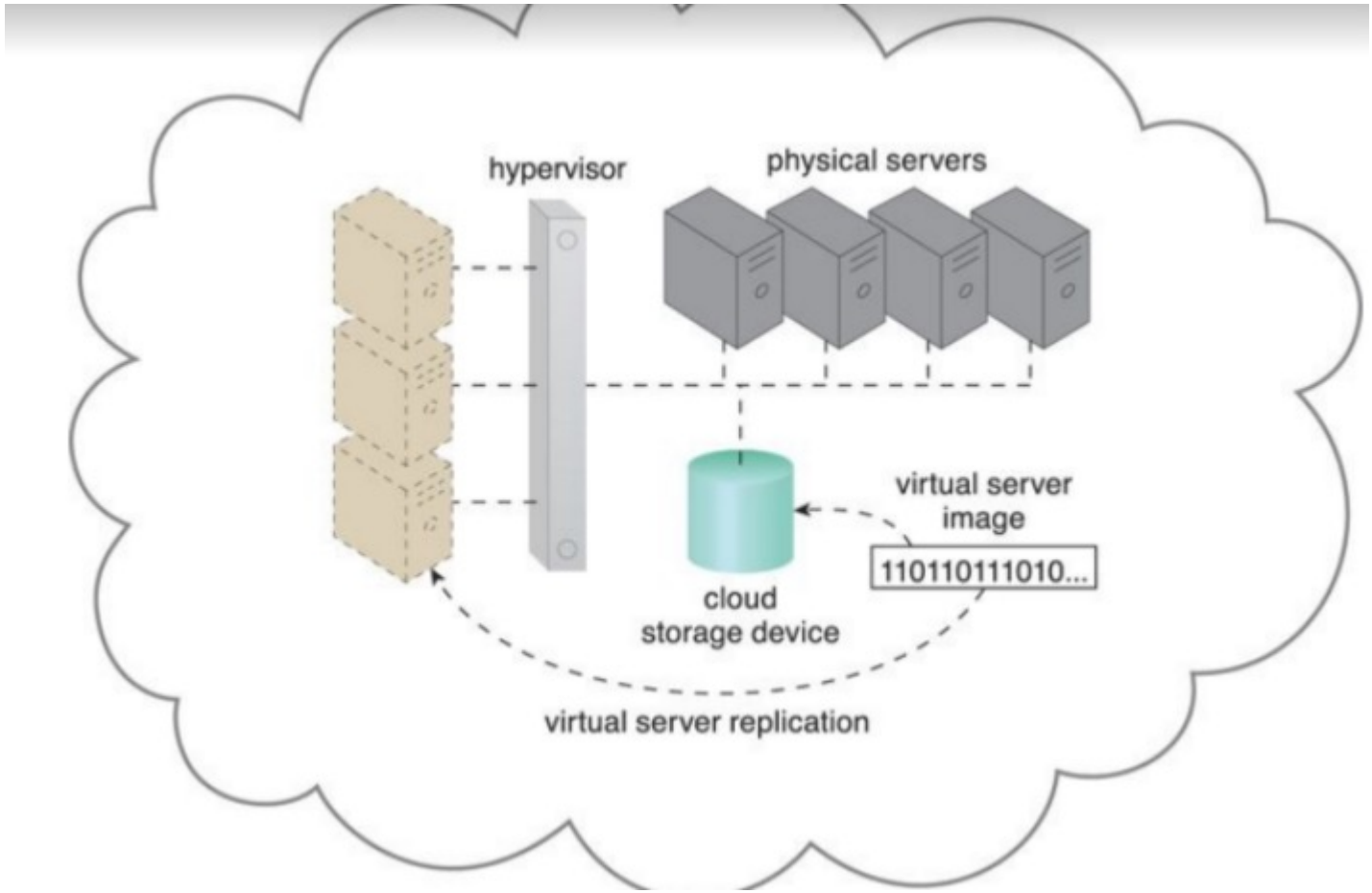


A polling agent monitors the status of a cloud service hosted by a virtual server by sending periodic polling request messages and receiving polling response messages that report usage status "A" after a number of polling cycles, until it receives a usage status of "B" (1), upon which the polling agent records the new usage status in the log database (2).

Resource Replication

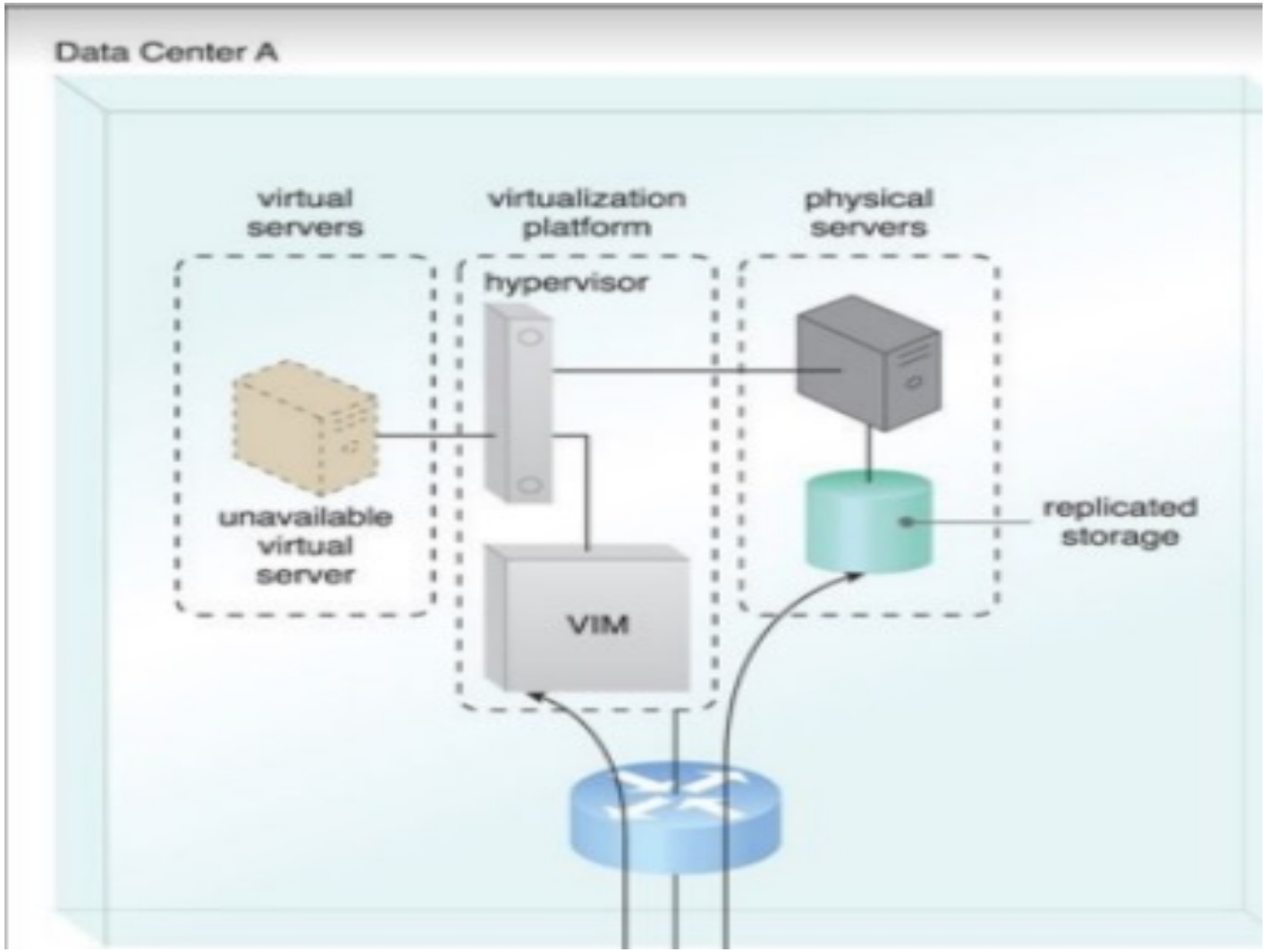
Defined as the creation of multiple instances of the same IT resource, replication is typically performed when an IT resource's *availability and performance need to be enhanced.*

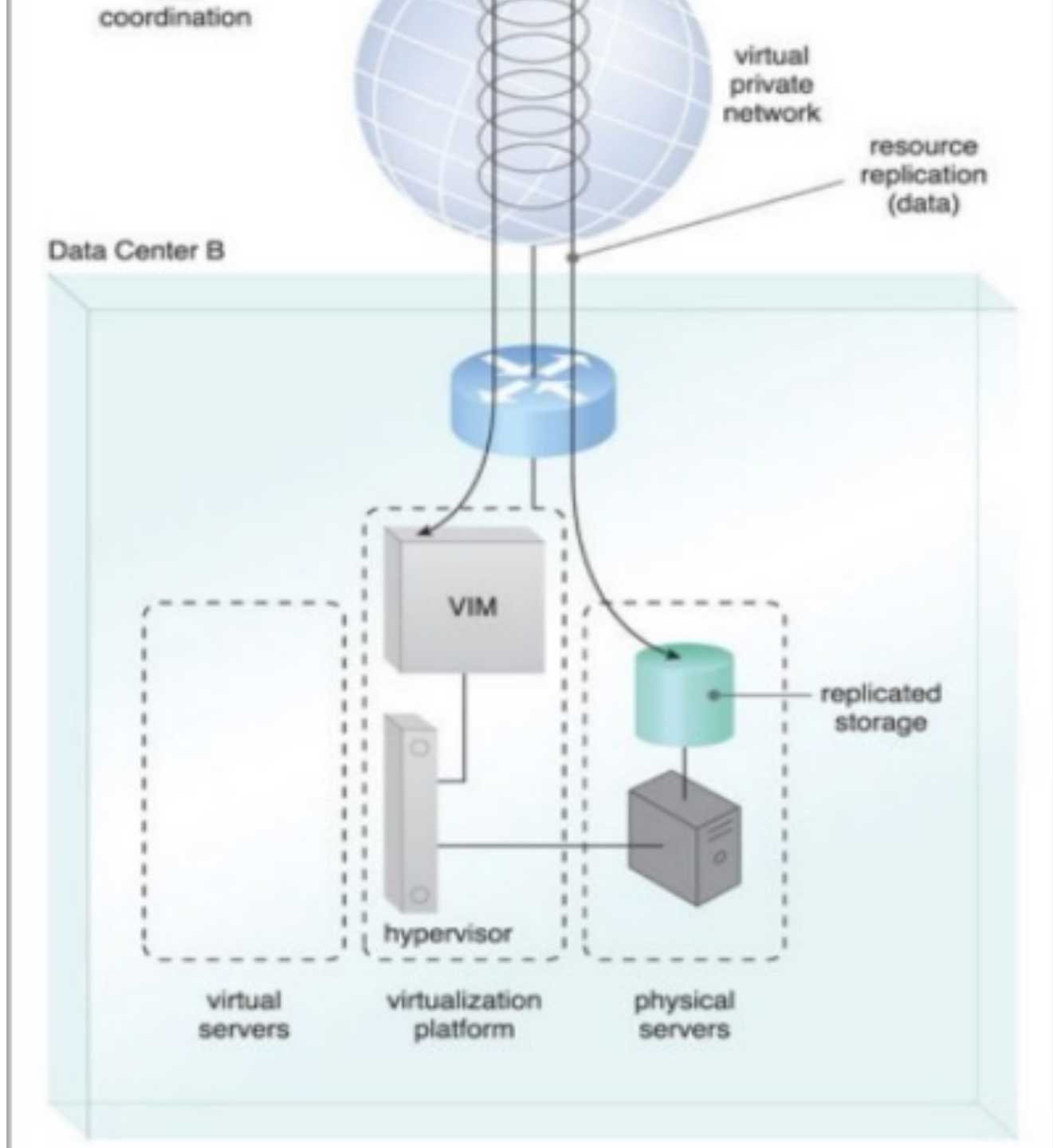
Virtualization technology is used to implement the resource replication mechanism to replicate cloud-based IT resources.



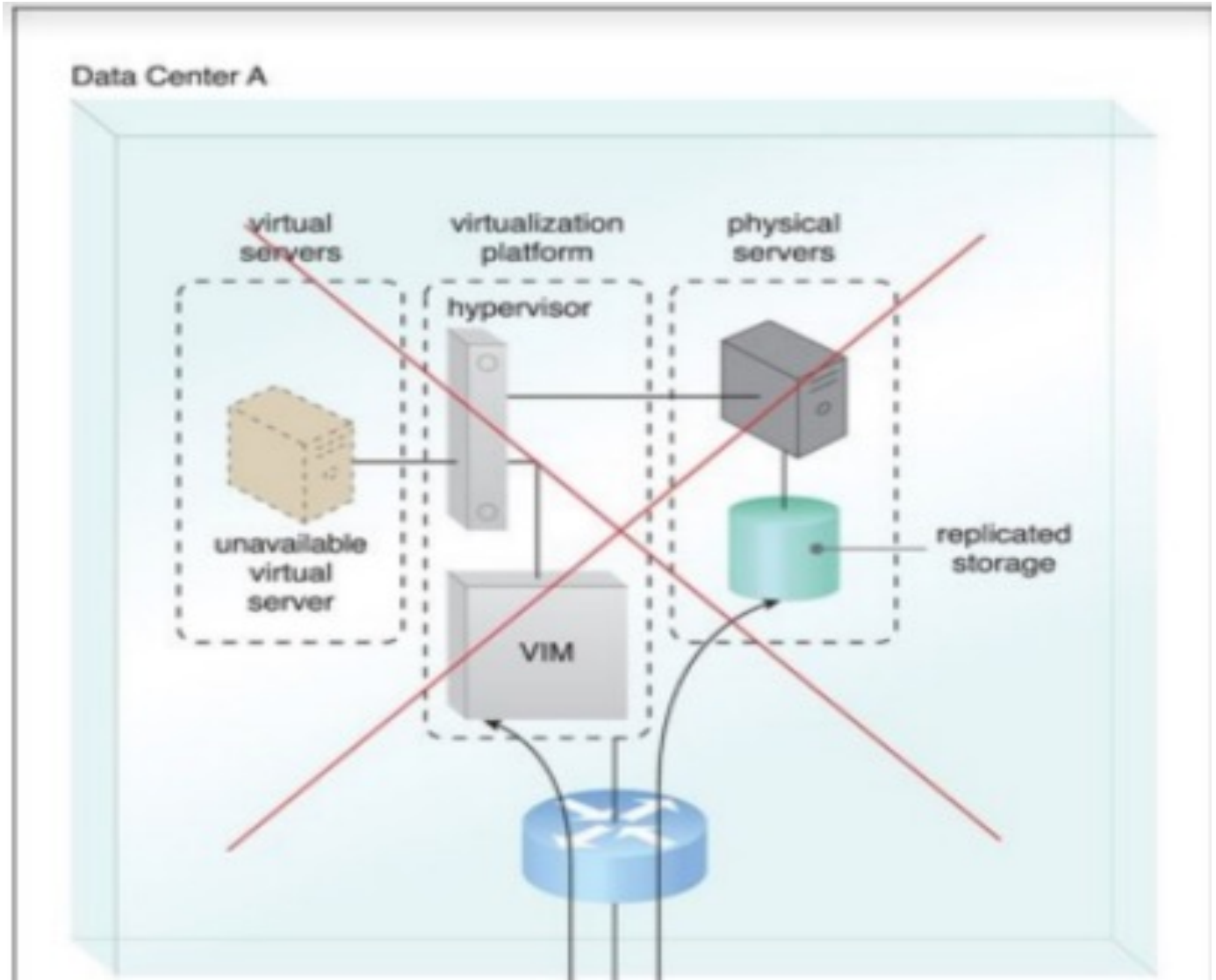
The hypervisor replicates several instances of a virtual server, using a stored virtual server image.

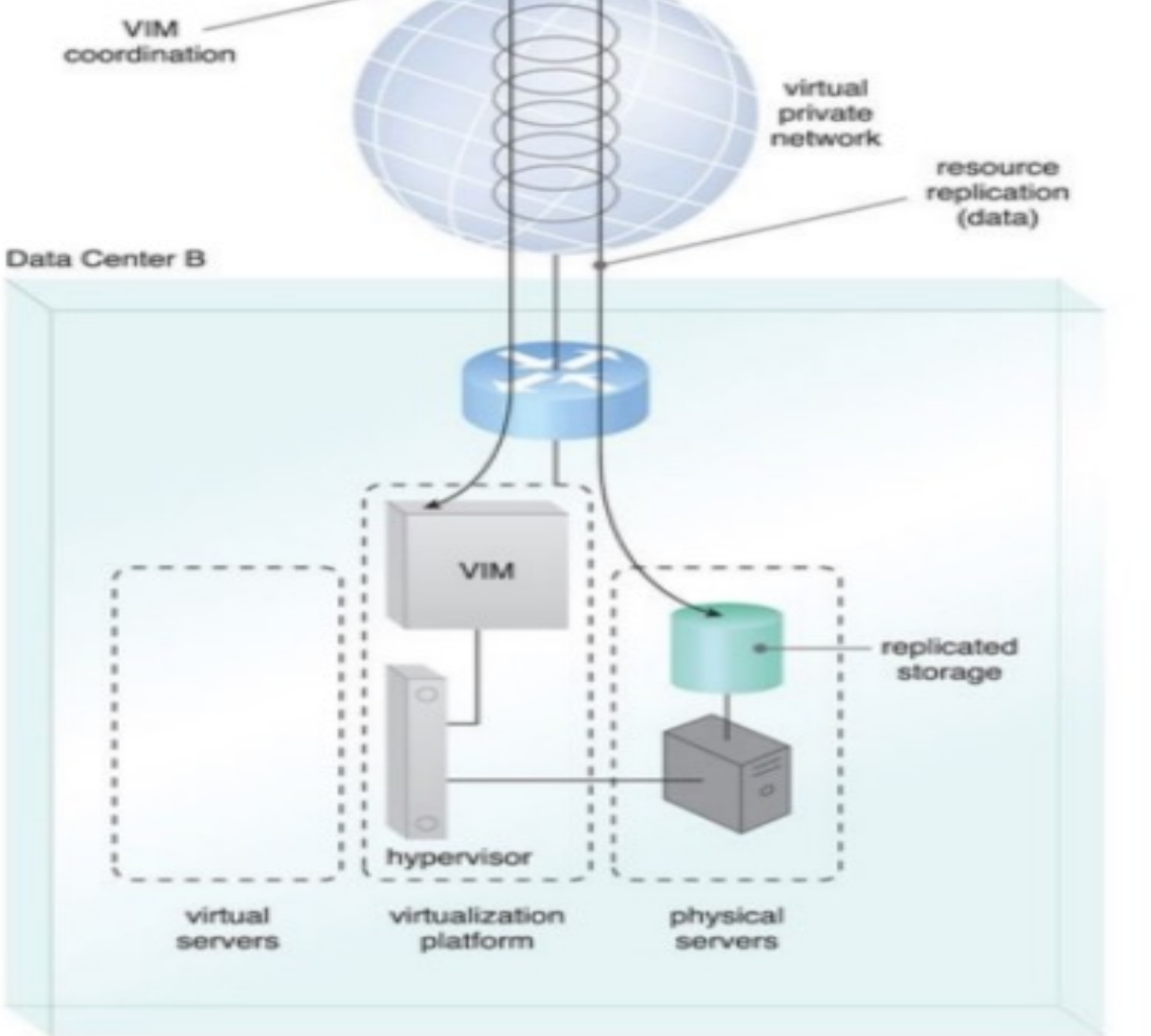
An example:A high-availability virtual server is running in Data Center A. VIM instances in Data Centers A and B are executing a coordination function that allows detection of failure conditions. Stored VM images are replicated between data centers as a result of the high-availability architecture.



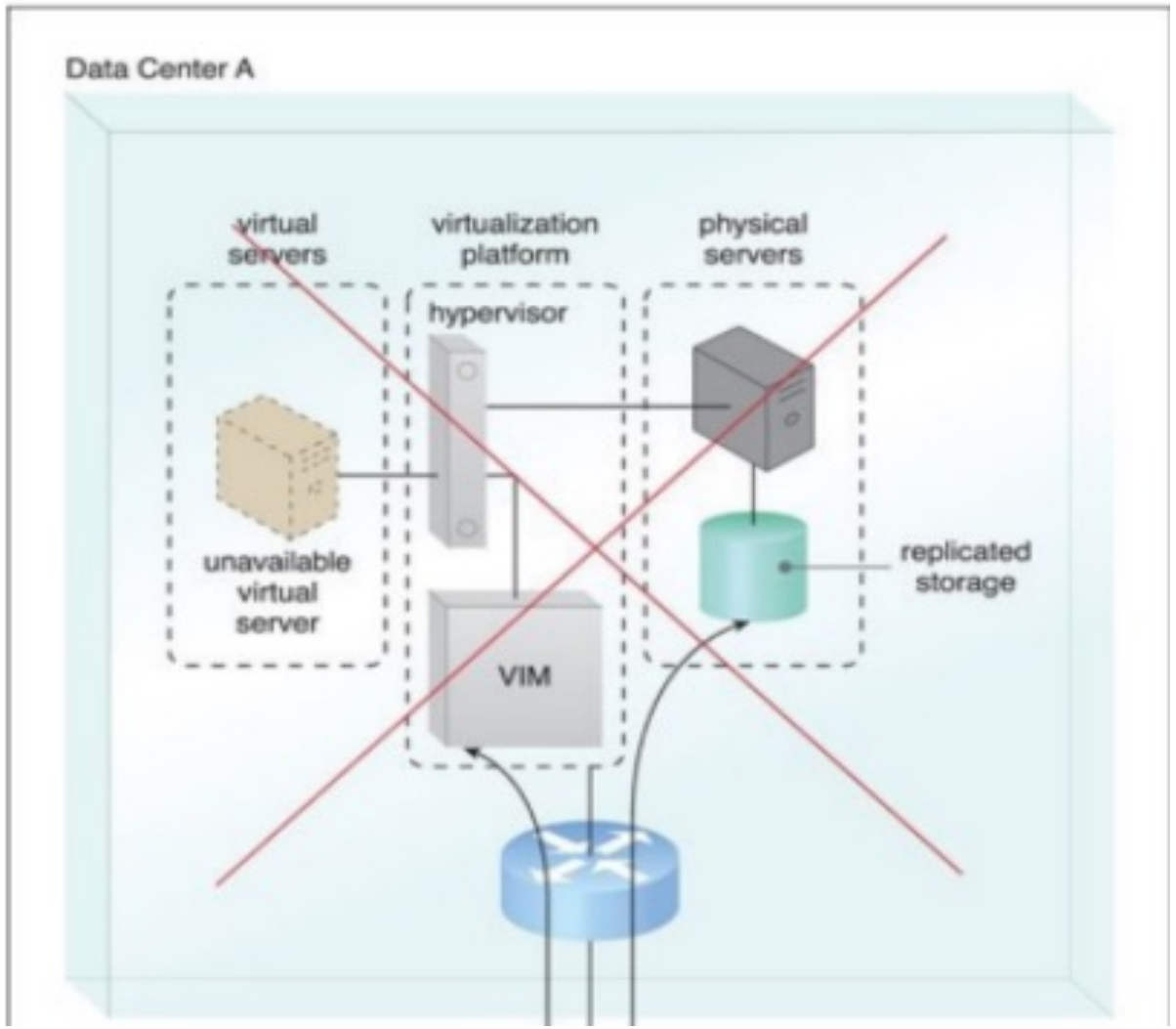

coordination
virtual
private
network
resource
replication
(data)
Data Center B
VIM
replicated
storage
hypervisor
virtual
servers
virtualization
platform
physical
servers

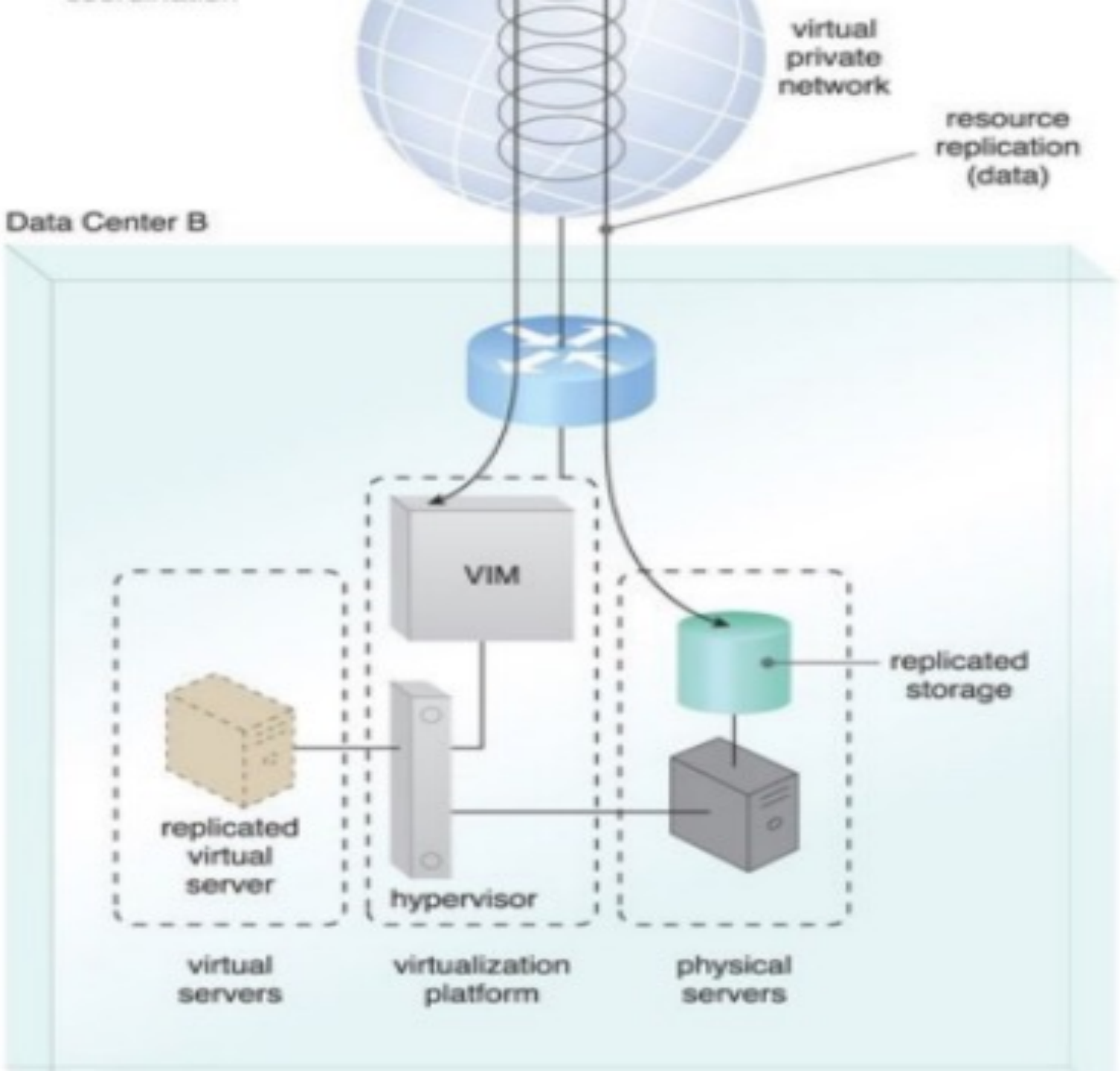
Case 2:The virtual server becomes unavailable in Data Center A. The VIM in Data Center B detects the failure condition and starts to reallocate the high-availability server from Data Center A to Data Center B.




VIM
coordination
virtual
private
network
resource
replication
(data)
Data Center B
VIM
replicated
storage
hypervisor
virtual
servers
virtualization
platform
physical
servers

Case 3: A new instance of the virtual server is created and made available in Data Center B.



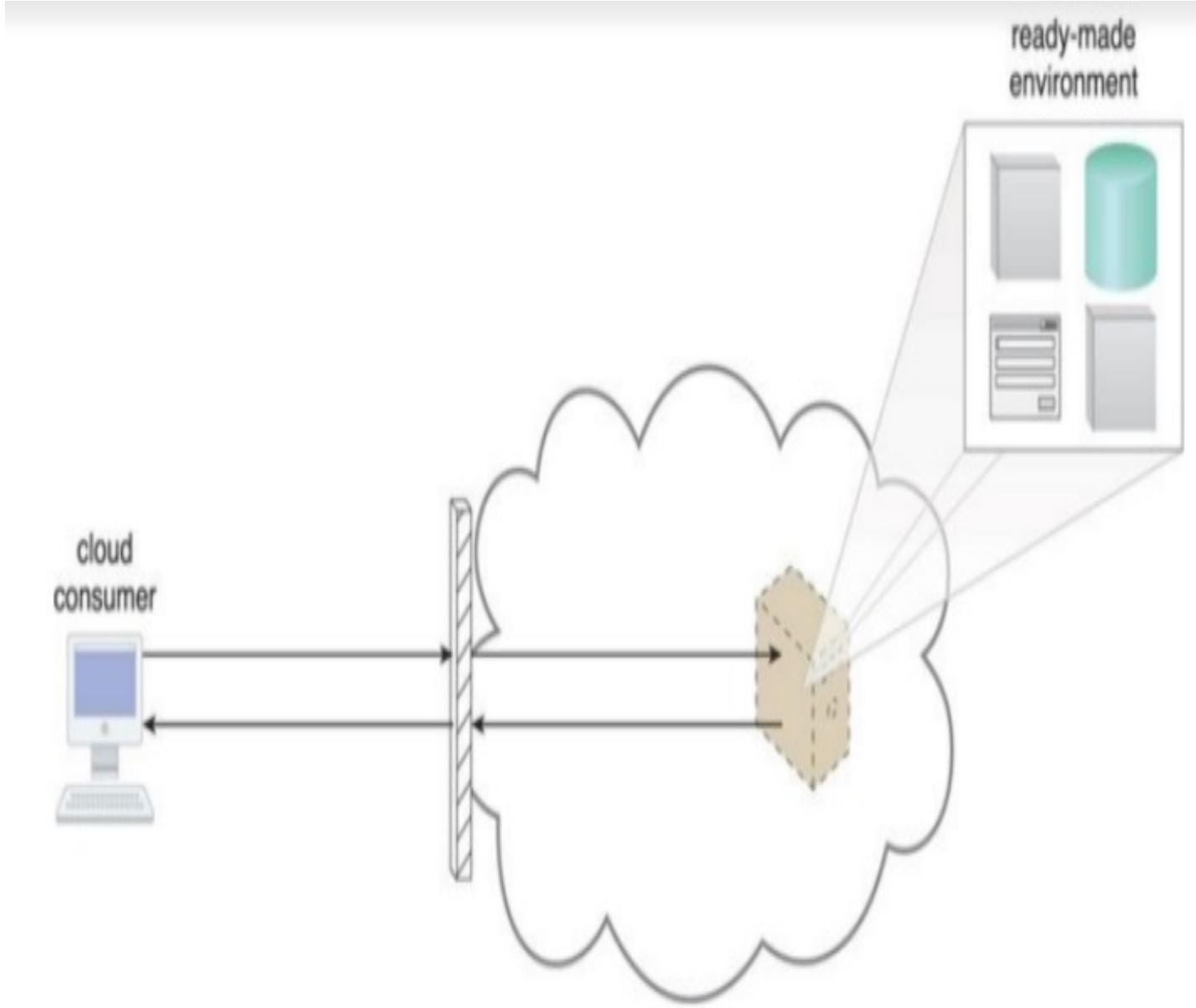

virtual
private
network
resource
replication
(data)
Data Center B
VIM
replicated
storage
replicated
virtual
server
hypervisor
virtual
servers
virtualization
platform
physical
servers

Ready-Made Environment

The ready-made environment mechanism is a defining component of the PaaS cloud delivery model that represents a pre-defined, cloud-based platform comprised of a set of already installed IT resources, ready to be used and customized by a cloud consumer.

These environments are utilized by cloud consumers to remotely develop and deploy their own services and applications within a cloud.

Typical ready-made environments include pre-installed IT resources, such as databases, middleware, development tools, and governance tools.



A cloud consumer accesses a ready-made environment hosted on a virtual server.

→A ready-made environment is generally equipped with a complete software development kit (SDK) that provides cloud consumers with programmatic access to the development technologies that comprise their preferred programming stacks.

→Middleware is available for multitenant platforms to support the development and deployment of Web applications.

→Some cloud providers offer runtime execution environments for cloud services that are based on different runtime performance and billing parameters.

→For example, a front-end instance of a cloud service can be configured to respond to time-sensitive requests more effectively than a back-end instance. The former variation will be billed at a different rate than the latter.

Automated Scaling Listener

The automated scaling listener mechanism is a service agent that monitors and tracks communications between cloud service consumers and cloud services for dynamic scaling purposes.

Automated scaling listeners are deployed within the cloud, typically near the firewall, from where they automatically track workload status information.

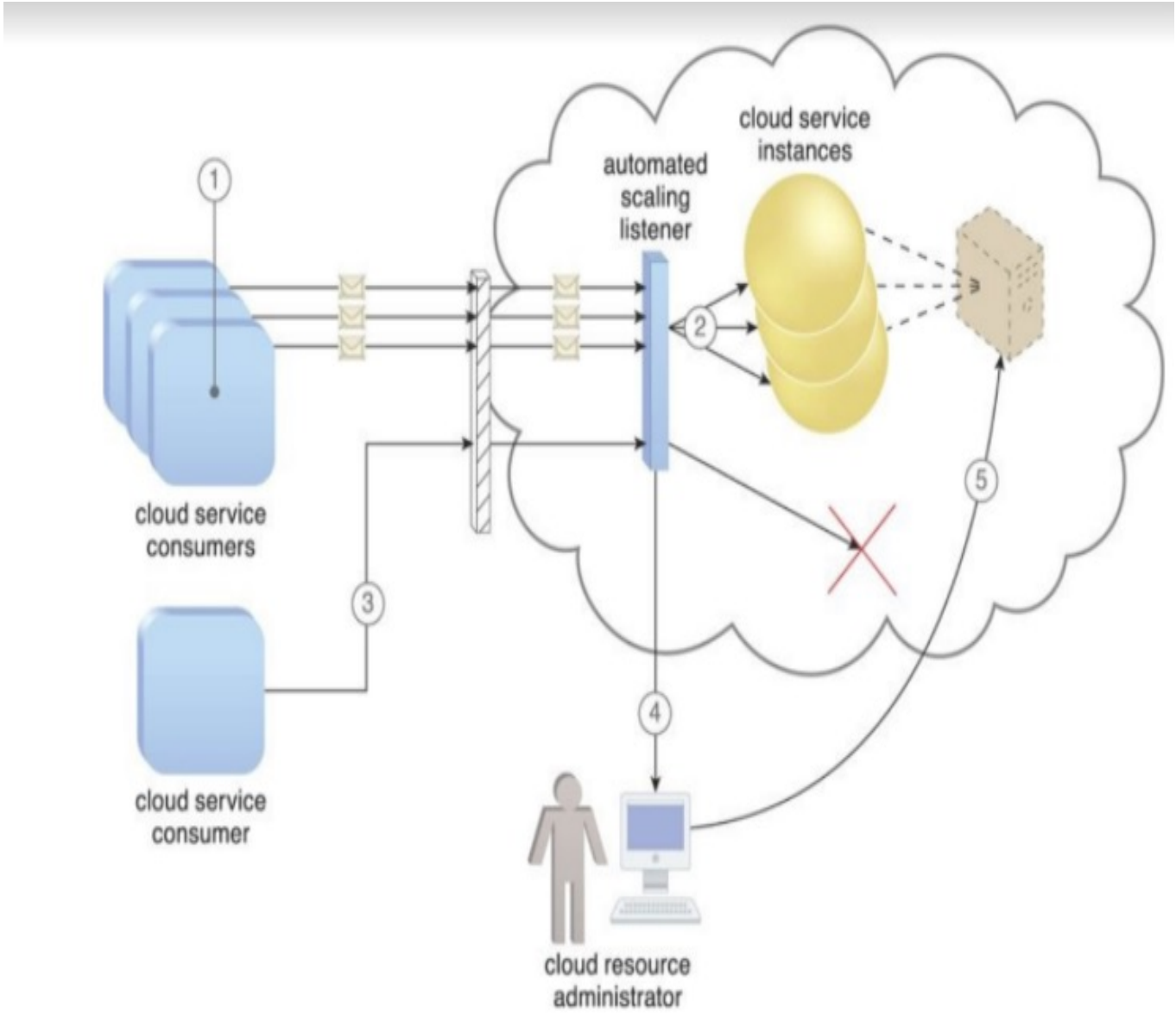
Workloads can be determined by the volume of cloud consumer-generated requests or via back-end processing demands triggered by certain types of requests.

For example, a small amount of incoming data can result in a large amount of processing.

- Automated scaling listeners can provide different types of responses to workload fluctuation conditions, such as:-

- Automatically scaling IT resources out or in based on parameters previously defined by the cloud consumer (commonly referred to as auto- scaling).

- Automatic notification of the cloud consumer when workloads exceed current thresholds or fall below allocated resources (Figure in next slide). This way,the cloud consumer can choose to adjust its current IT resource allocation.



Three cloud service consumers attempt to access one cloud service simultaneously (1). The automated scaling listener scales out and initiates the creation of three redundant instances of the service (2). A fourth cloud service consumer attempts to use the cloud service (3). Programmed to allow up to only three instances of the cloud service, the automated scaling listener rejects the fourth attempt and notifies the cloud consumer that the requested workload limit has been exceeded (4). The cloud consumer's cloud resource administrator accesses the remote administration environment to adjust the provisioning setup and increase the redundant instance limit (5). Different cloud provider vendors have different names for service agents that act as automated scaling listeners.

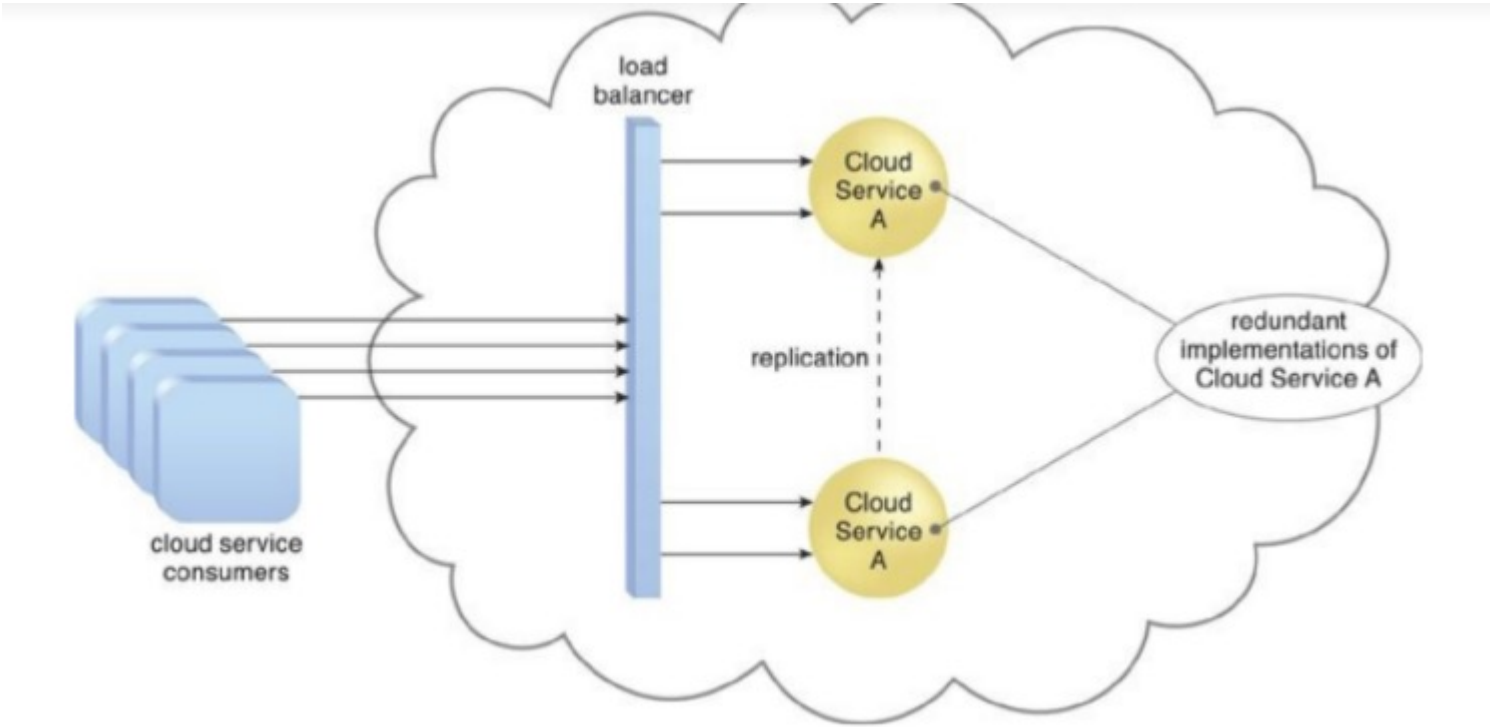
Load Balancer

- A common approach to horizontal scaling is to balance a workload across two or more IT resources to increase performance and capacity beyond what a single IT resource can provide. The load balancer mechanism is a runtime agent with logic fundamentally based on this premise.
- Beyond simple division of labor algorithms (Figure next slide), load balancers can perform a range of specialized runtime workload distribution functions that include:-

Asymmetric Distribution: – larger workloads are issued to IT resources with higher processing capacities.

Workload Prioritization – workloads are scheduled, queued, discarded, and distributed workloads according to their priority levels.

Content-Aware Distribution – requests are distributed to different IT resources as dictated by the request content.



A load balancer implemented as a service agent transparently distributes incoming workload request messages across two redundant cloud service implementations, which in turn **maximizes performance for the cloud service consumers.**

A load balancer is programmed or configured with a set of performance and QoS rules and parameters with the general objectives of optimizing IT resource usage, avoiding overloads, and maximizing throughput.

The load balancer mechanisms can exist as a:

- Multi-layer network switch
- Dedicated hardware appliance
- Dedicated software-based system (common in server operating systems)
- Service agent (usually controlled by cloud management software)

The load balancer is typically located on the communication path between the IT resources generating the workload and the IT resources performing the workload processing.

This mechanism can be designed as a **transparent agent that remains hidden from the cloud service consumers, or as a proxy component that abstracts the IT resources performing their workload.**

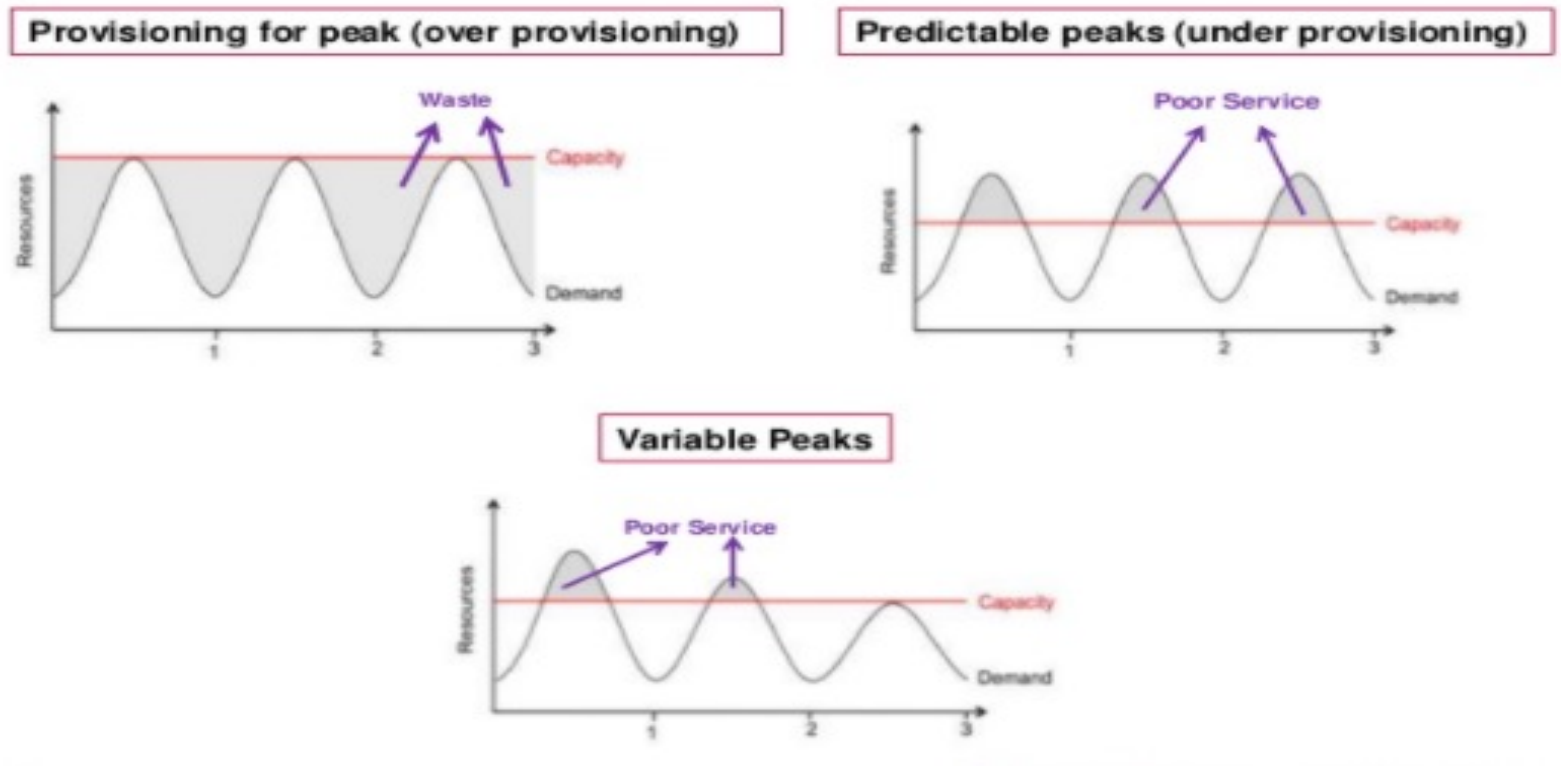
SLA and SLA Monitoring 413

- SLOs:-Service-level objectives

Example: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.

Conditions of over provision and under provisioning



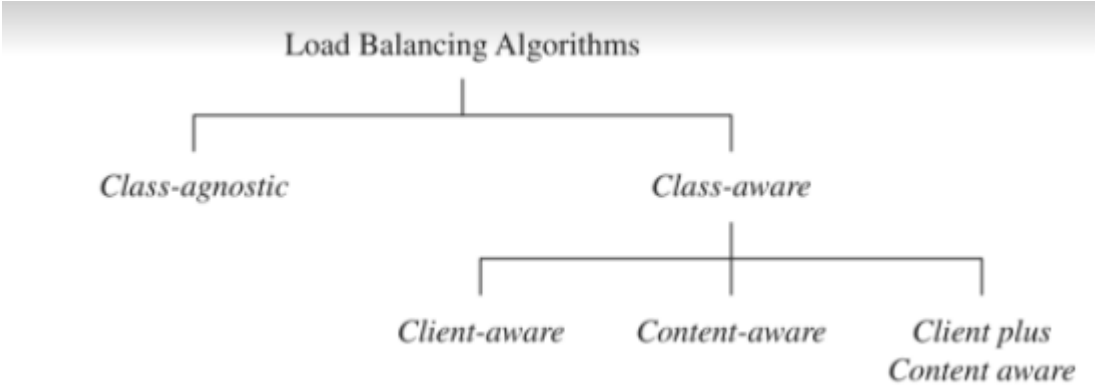
https://www.slideshare.net/dkosaraju/agility-and-cloud-computing-voices-2015

- To fulfill the SLOs mentioned in the application SLA and also make their IT infrastructure elastic, an in-depth understanding of the application's behaviour is required for the MSP (Managed Service Provider).

TRADITIONAL APPROACHES TO SLO MANAGEMENT

- Load Balancing (covered before)

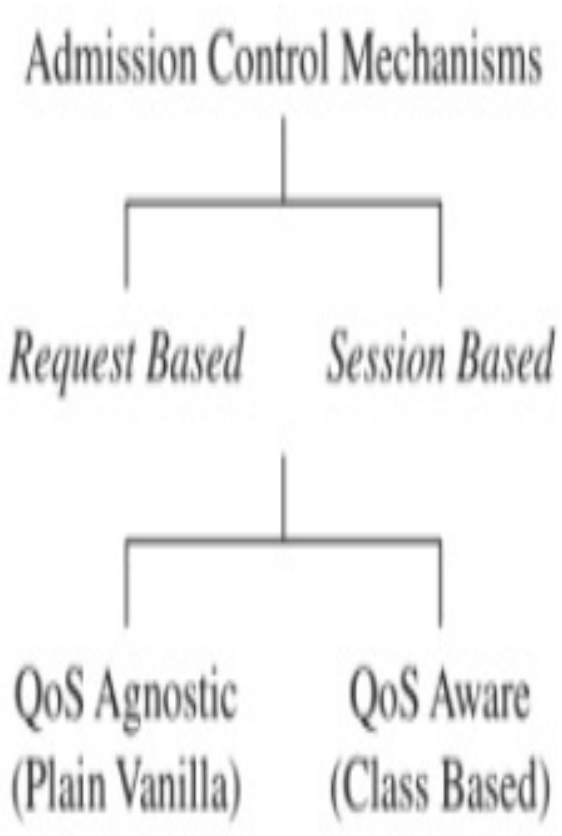
Taxonomy of the load balancing:-



Load balancing taxonomy from cloud computing : principle and paradigm book by Raj Kumar Buyya

Admission Control

- Admission control algorithms play an important role in deciding the set of requests that should be admitted into the application server when the server experiences “very” heavy loads.
- The objective of admission control mechanisms, therefore, is to police the incoming requests and identify a subset of incoming requests that can be admitted into the system when the system faces overload situations.



Request-based admission control algorithms reject new requests if the servers are running to their capacity. The disadvantage with this approach is that a client's session may consist of multiple requests that are not necessarily unrelated. **session-based admission control mechanisms** try to ensure that longer sessions are completed and any new sessions are rejected. Accordingly, once a session is admitted into the server, all future requests belonging to that session are admitted as well, even though new sessions are rejected by the system.

Furthermore, the decision to reject a request can depend on the type of user making the request or the nature of the request being made. For example, a new request or a new session initiated by a ***high-priority user may be admitted while the requests from low- priority users are rejected***. Similarly, requests that are likely to consume more system resources can be rejected during overload situations. Such admission control mechanisms are called QoS-aware control mechanisms.

TYPES OF SLA

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.

Infrastructure 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

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

LIFE CYCLE OF SLA 453

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

Contract Definition. Generally, service providers define a set of service offerings and corresponding SLAs using standard templates. These service offerings form a ***catalog***. Individual SLAs for enterprises can be derived by customizing these base SLA templates.

Publication and Discovery. Service provider advertises these base service offerings through standard publication media, and the customers should be able to locate ***the service provider by searching the catalog.*** The customers can search different competitive offerings and shortlist a few that fulfill their requirements for further negotiation.

- ***Negotiation.*** Once the customer has discovered a service provider who can meet their application hosting need, **the SLA terms and conditions needs to be mutually agreed upon before signing the agreement for hosting the application.**
- ***Operationalization***. SLA operation consists of SLA monitoring, SLA accounting, and SLA enforcement. SLA monitoring involves measuring parameter values and calculating the metrics defined as a part of SLA and determining the deviations.

De-commissioning. SLA decommissioning involves termination of all activities performed under a particular SLA when the hosting relationship between the service provider and the service consumer has ended. SLA specifies the terms and conditions of contract termination and specifies situations under which the relationship between a service provider and a service consumer can be considered to be legally ended.

SLA MANAGEMENT IN CLOUD

SLA management of applications hosted on cloud platforms involves **five phases.**

1. Feasibility
2. On-boarding
3. Pre-production
4. Production
5. Termination

Feasibility Analysis

- MSP conducts the feasibility study of hosting an application on their cloud platforms. This study involves three kinds of **feasibility: (1) technical feasibility, (2) infrastructure feasibility, and (3) financial feasibility.**

On-Boarding of Application

- Once the customer and the MSP agree in principle to host the application based on the findings of the feasibility study, the application is moved from the customer servers to the **hosting platform.**
- Moving an application to the MSP's hosting platform is called on-boarding

On-Boarding of Application: *Operational Policies*

defined. Operational policies (OP) are represented in the following format:

$$\text{OP} = \text{collection of } \langle \text{Condition, Action} \rangle$$

Here the action could be workflow defining the sequence of actions to be undertaken. For example, one OP is

$$\text{OP} = \langle \text{average latency of web server} > 0.8 \text{ sec, scale-out the web-server tier} \rangle$$

Preproduction

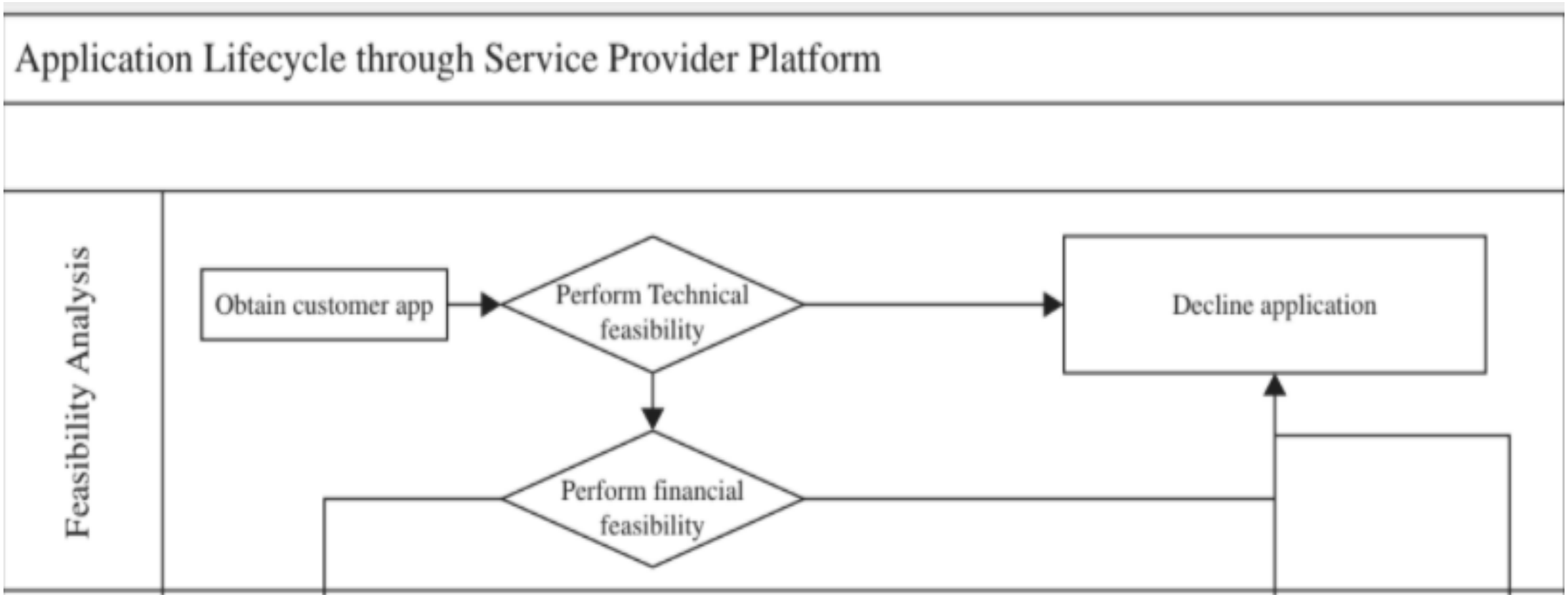
- Once the determination of policies is completed as discussed in previous phase, the application is hosted in a **simulated production environment.** It facilitates the customer to verify and validate the MSP's findings on application's runtime characteristics and agree on the defined SLA. Once both parties agree on the cost and the terms and conditions of the SLA, the customer sign-off is obtained. On successful completion of this phase the MSP allows the application to go on-live.

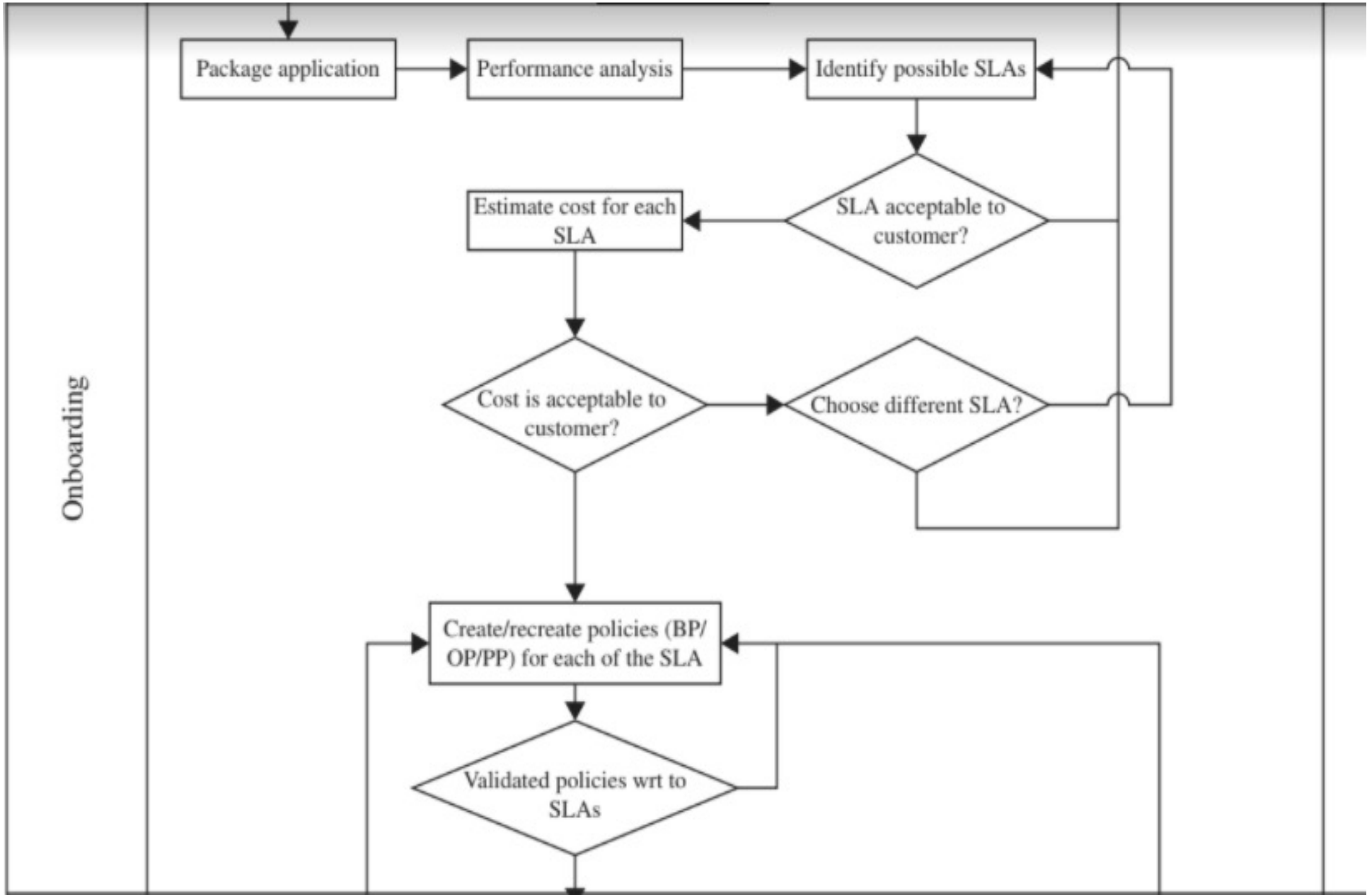
Production

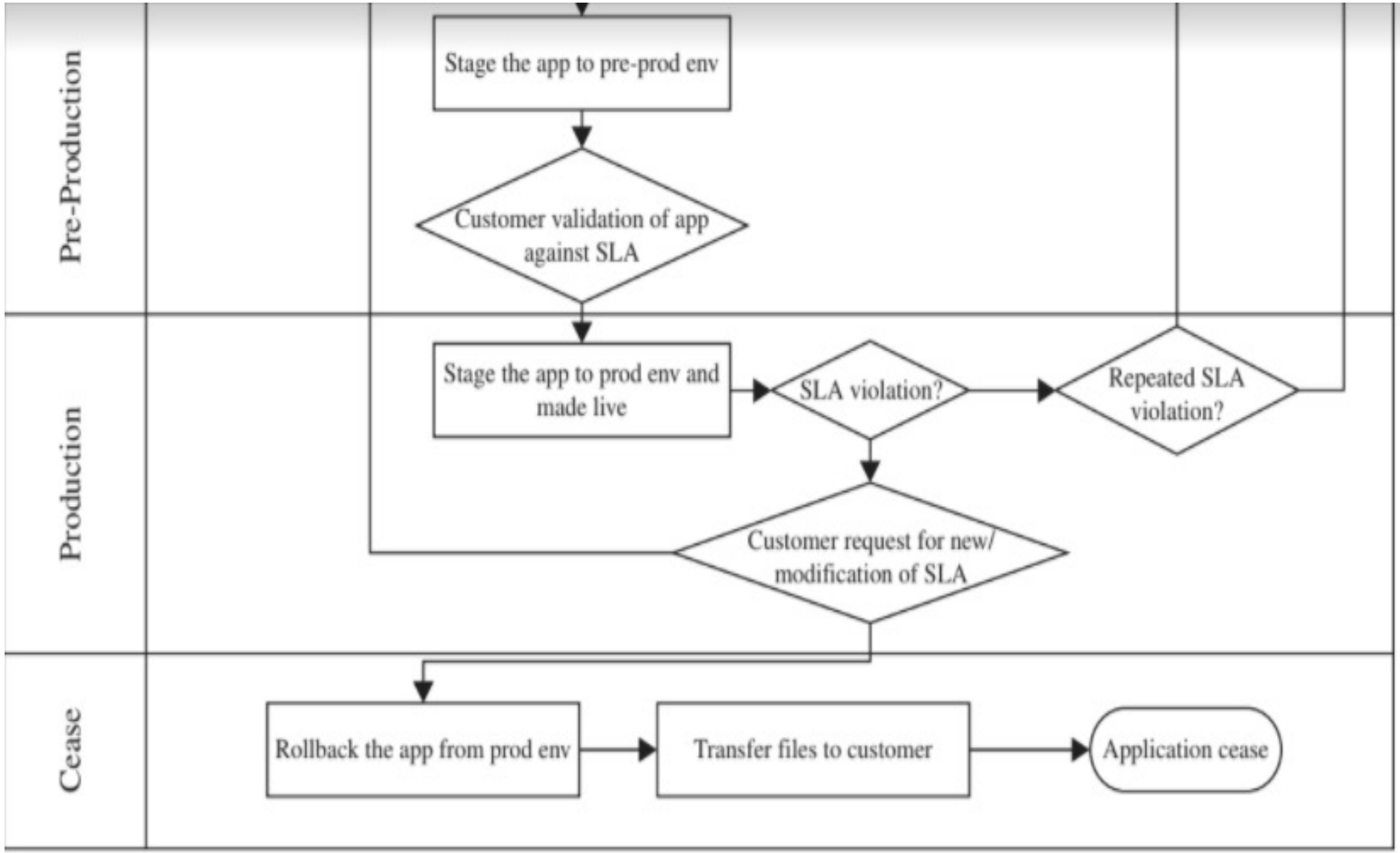
- In this phase, the application is made accessible to its end users under the agreed SLA. **However, there could be situations when the managed application tends to behave differently in a production environment compared to the preproduction environment**. This in turn may cause sustained breach of the terms and conditions mentioned in the SLA.

Termination

- **When the customer wishes to withdraw the hosted application and does not wish to continue to avail the services of the MSP for managing the hosting of its application, the termination activity is initiated.** On initiation of termination, all data related to the application are transferred to the customer and only the essential information is retained for legal compliance. This ends the hosting relationship between the two parties for that application, and the customer sign-off is obtained.


Application Lifecycle through Service Provider Platform
Feasibility Analysis
Obtain customer app
Perform Technical feasibility
Decline application
Perform financial feasibility


Onboarding
Package application
Performance analysis
Identify possible SLAs
SLA acceptable to customer?
Estimate cost for each SLA
Cost is acceptable to customer?
Choose different SLA?
Create/recreate policies (BP/ OP/PP) for each of the SLA
Validated policies wrt to SLAs


Pre-Production
Production
Cease
Stage the app to pre-prod env
Customer validation of app against SLA
Stage the app to prod env and made live
SLA violation?
Repeated SLA violation?
Customer request for new/ modification of SLA
Rollback the app from prod env
Transfer files to customer
Application cease

AUTOMATED POLICY-BASED MANAGEMENT

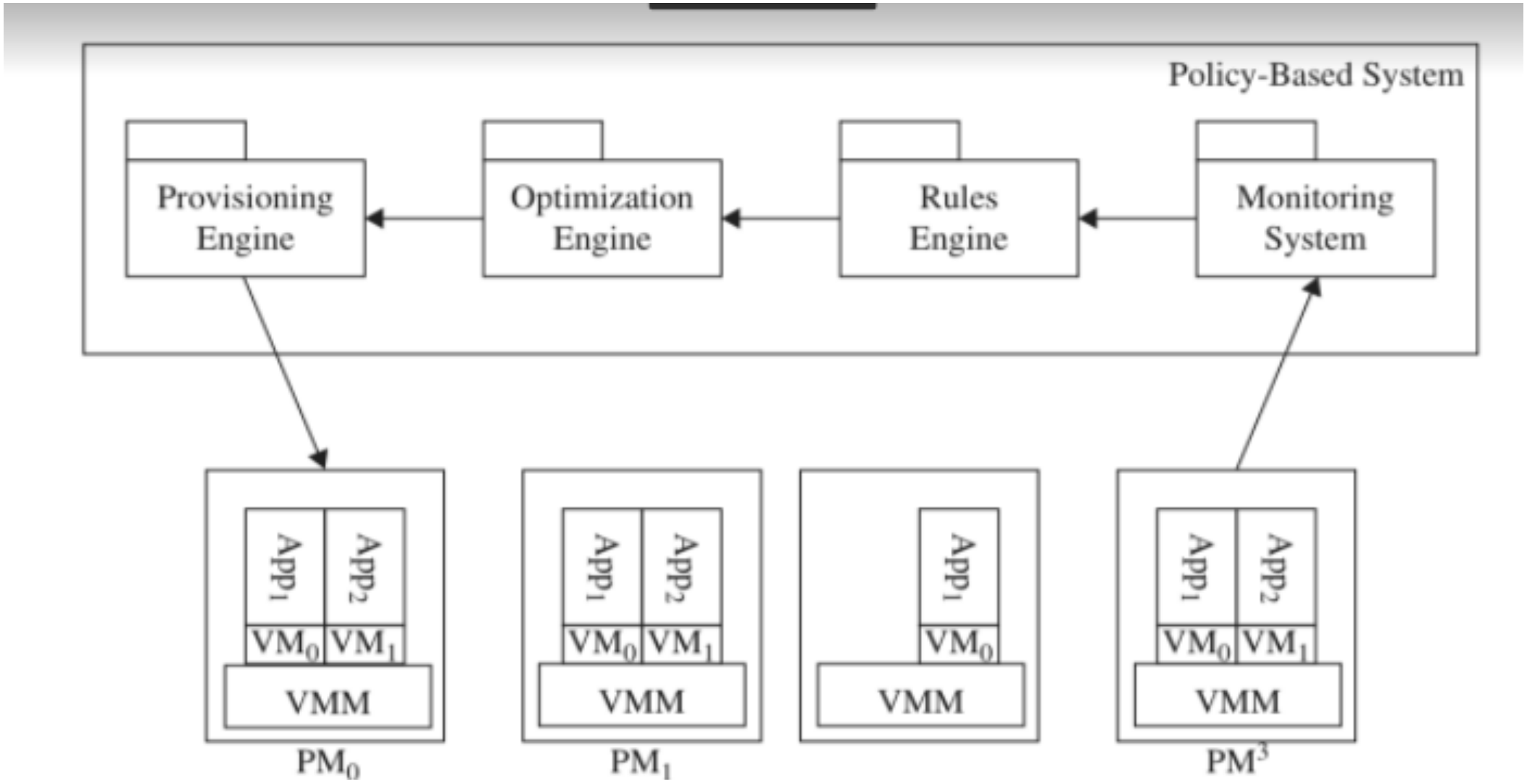
- *Operational policies* specify the functional relationship between the system-level infrastructural attributes and the business level SLA goals. Knowledge of such a relationship helps in identifying the quantum of system resources to be allocated to the various components of the application for different system attributes at various workloads, workload compositions, and operating conditions so that the SLA goals are met.

Some of the parameters often used to prioritize action and perform resource contention resolution are:

- 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.

- **Priority ranking algorithms** use these parameters to derive **scores.**
- These scores are used to rank each of the actions that contend for the same resources.
- **Actions having high scores get higher priority and hence, receive access to the contended resources.**

Importance of optimization in the policy-based management system.



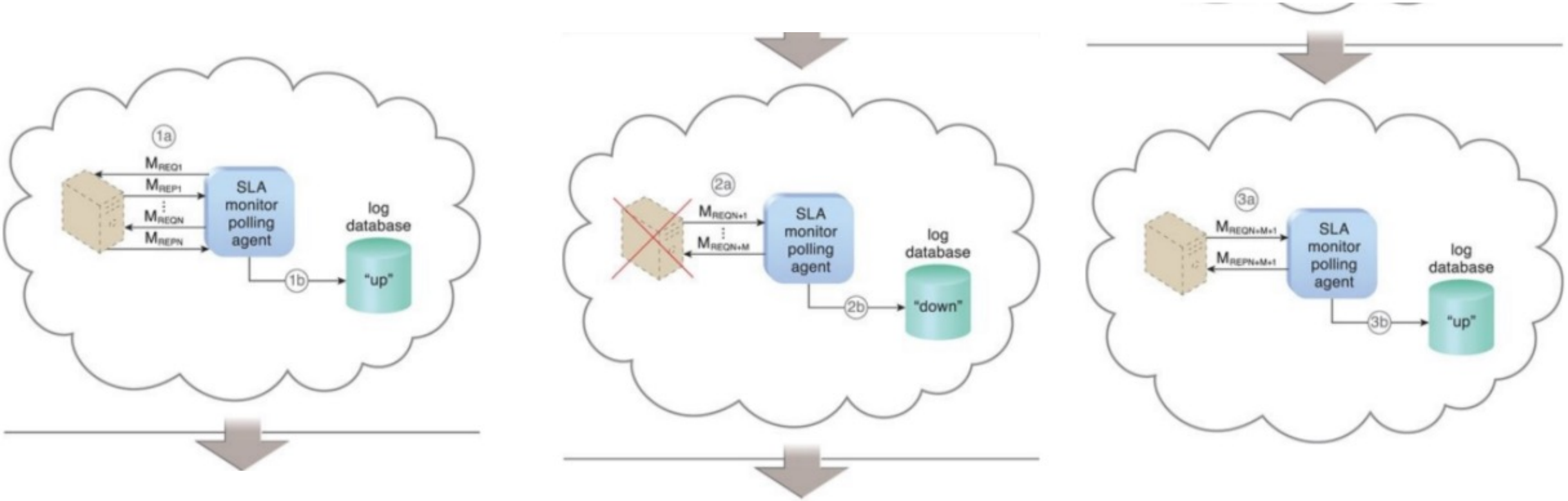
- ***Prioritization Engine.*** Requests from different customers' web applications contending for the same resource are identified, and accordingly their execution is prioritized. Business policies defined by the MSP helps in identifying the requests **whose execution should be prioritized** in case of resource contentions so that the MSP can realize higher benefits.
- ***Provisioning Engine.*** Every user request of an application will be enacted by the system. The set of steps necessary to enact the user requests are defined in the **provisioning policy, and they are used to fulfill the application request like starting an application, stopping an application, and so on**. This set of steps can be visualized as a workflow. Hence, the execution of the provisioning policy requires a **workflow engine**.

- ***Rules Engine.*** The ***operation policy*** defines a sequence of actions to be enacted under different conditions/trigger points. The rules engine evaluates the data captured by the monitoring system, evaluates against the predefined operation rules, and triggers the associated action if required. ***Rules engine and the operational policy is the key to guaranteeing SLA under a self healing system.***
- ***Monitoring System***. Monitoring system collects the defined metrics in SLA. These metrics are used for monitoring resource failures, evaluating operational policies, and auditing and billing purpose.

- ***Auditing.*** The adherence to the predefined **SLA needs to be monitored and recorded.** It is essential to monitor the compliance of SLA because any noncompliance leads to strict penalties. The audit report forms the basis for strategizing and long-term planning for the MSP.
- ***Accounting/Billing System***. Based on the payment model, chargebacks could be made based on the **resource utilized by the process during the operation. The fixed cost and recurring costs are computed and billed accordingly.**

SLA Monitor

- The SLA monitor mechanism is used to specifically observe the runtime performance of cloud services to ensure that they are fulfilling the contractual QoS requirements that are published in SLAs (next page).
- The data collected by the SLA monitor is processed by an SLA management system to be aggregated into SLA reporting metrics.
- The system can proactively repair or failover cloud services when exception conditions occur, such as when the SLA monitor reports a cloud service as “down.”



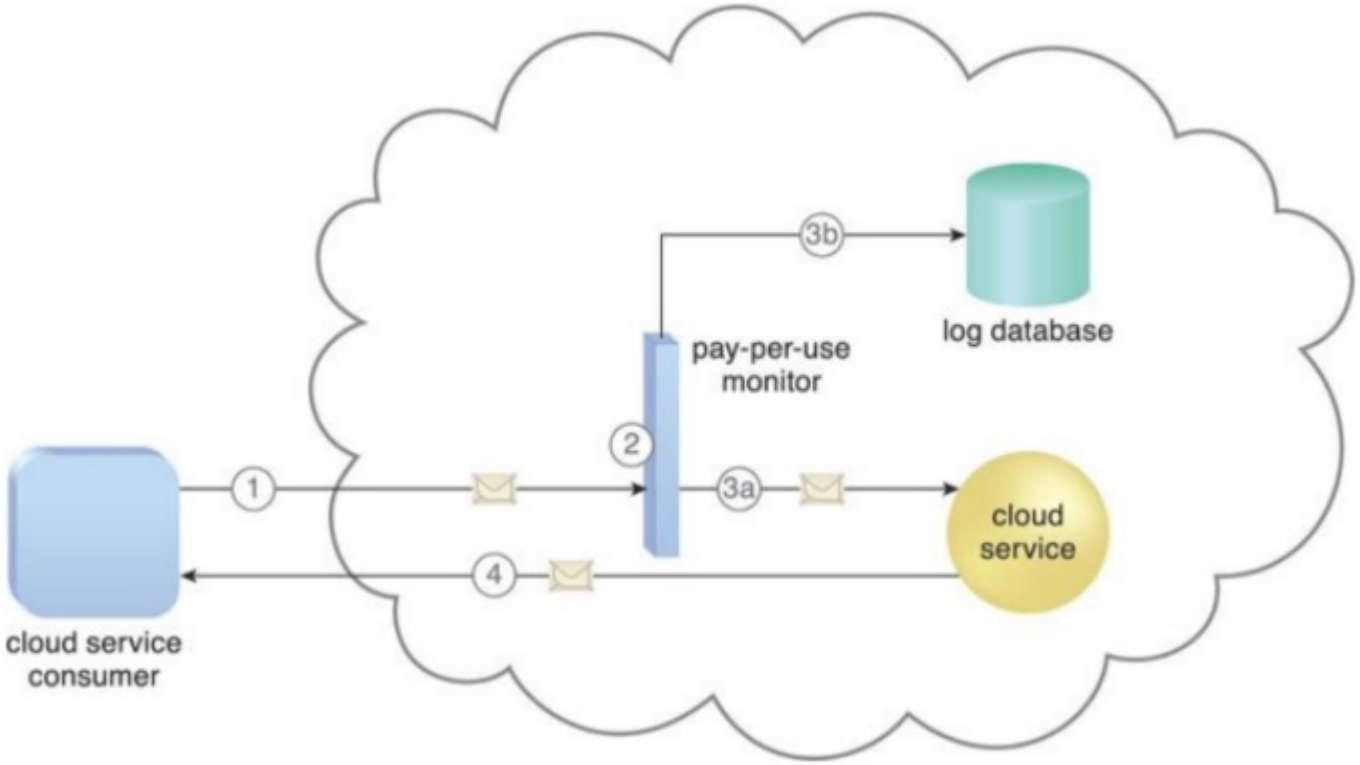
The SLA monitor polls the cloud service by sending over polling request messages (MREQ1 to MREQN). The monitor receives polling response

messages (MREP1 to MREPN) that report that the service was "up" at each polling cycle (1a). The SLA monitor stores the "up" time—time period of all polling

cycles 1 to N—in the log database (1b).

- The SLA monitor polls the cloud service that sends polling request messages (MREQN+1 to MREQN+M). Polling response messages are not received (2a). The response messages continue to time out, so the SLA monitor stores the "**down**" time—time period of all polling cycles N+1 to N+M—in the log database (2b).
- The SLA monitor sends a polling request message (MREQN+M+1) and receives the polling response message (MREPN+M+1) (3a). The SLA monitor stores the "**up**" time in the log database (3b).

Another case study

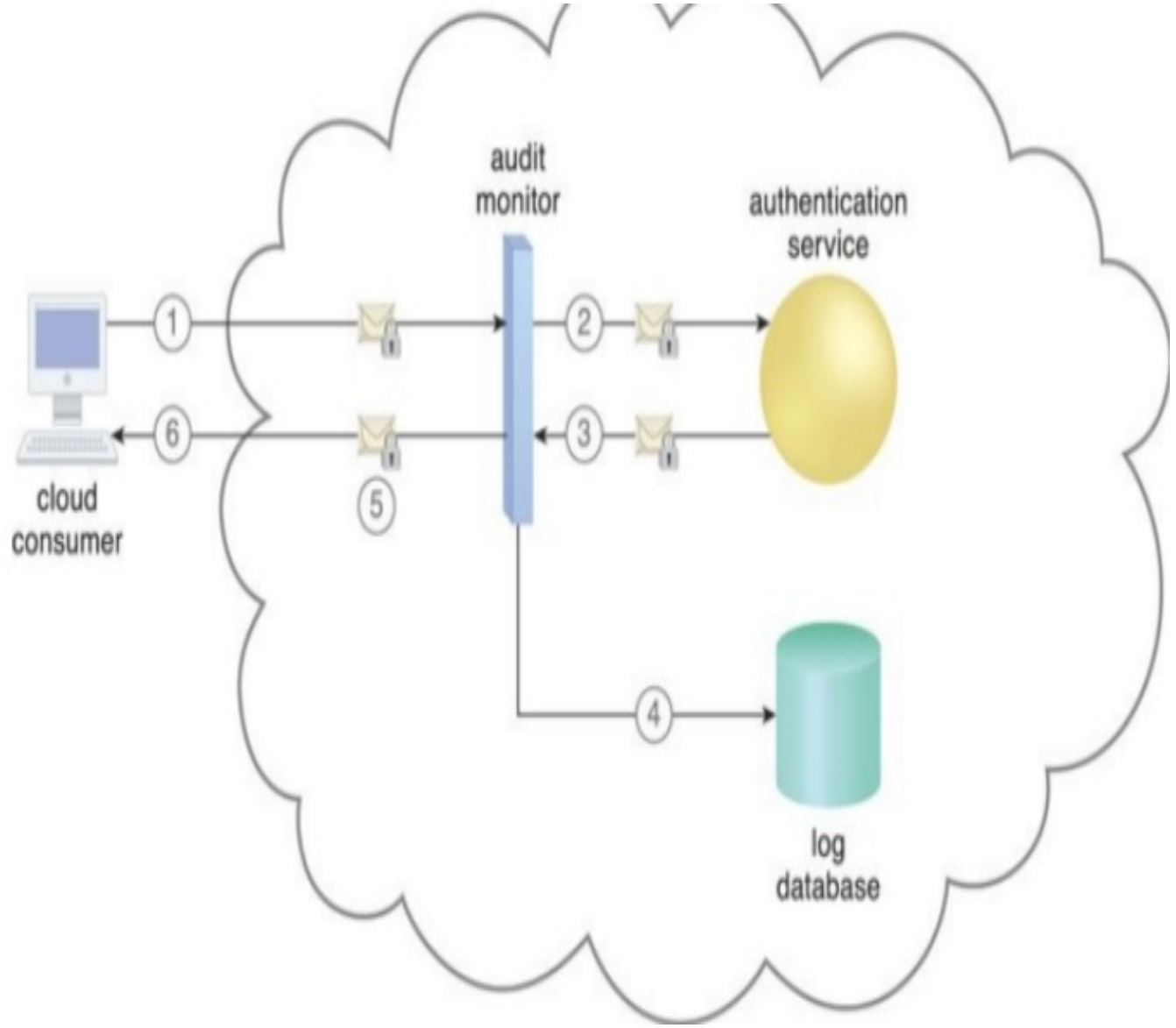


A cloud service consumer sends a request message to the cloud
service (1).
The pay-per-use monitor intercepts the message (2),
forwards it to the cloud service (3a),
and stores the usage information in accordance with its monitoring metrics
(3b).
The cloud service forwards the response messages back to the cloud service consumer to provide the requested service (4).

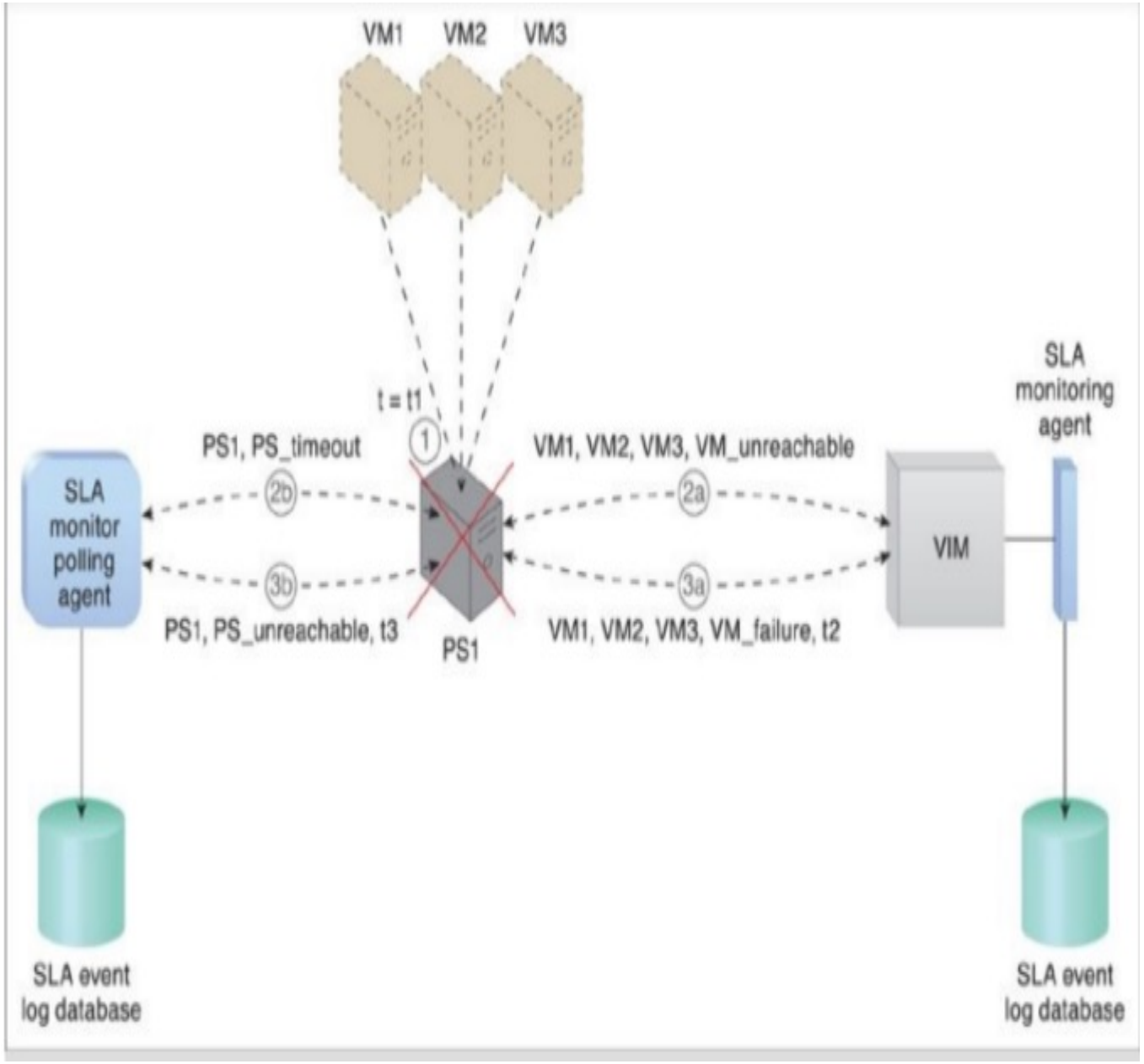
Audit Monitor

The audit monitor mechanism is used to **collect audit tracking data for networks and IT resources** in support of (or dictated by) regulatory and contractual obligations.

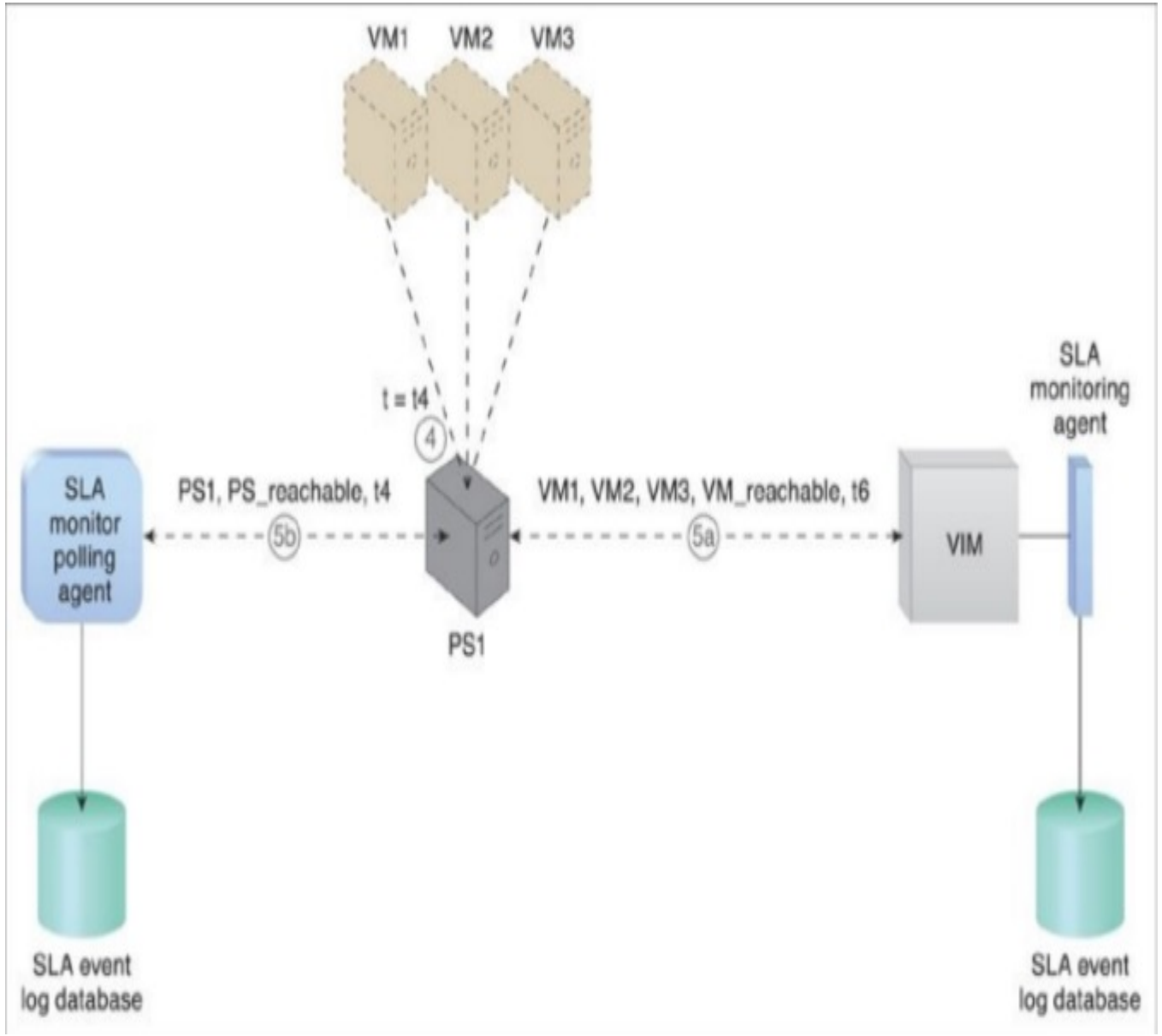
Figure in next slide depicts an audit monitor implemented as a monitoring agent that intercepts "login" requests and stores the requestor's security credentials, as well as both failed and successful login attempts, in a log database for future audit reporting purposes.



A cloud service consumer requests access to a cloud service by sending a login request message with security credentials (1). The audit monitor intercepts the message (2) and forwards it to the authentication service (3). The authentication service processes the security credentials. A response message is generated for the cloud service consumer, in addition to the results from the login attempt (4). The audit monitor intercepts the response message and stores the entire collected login event details in the log database, as per the organization's audit policy requirements (5). Access has been granted, and a response is sent back to the cloud service consumer (6).



At timestamp = t1, the physical host server has failed and becomes unavailable (1). The SLA monitoring agent captures a VM_unreachable event that is generated for each virtual server in the failed host server (2a). The SLA monitor polling agent stops receiving responses from the host server and issues PS_timeout events (2b). At timestamp = t2, the SLA monitoring agent captures a VM_failure event that is generated for each of the failed host server's three virtual servers (3a). The SLA monitor polling agent starts to issue PS_unavailable events after three successive PS_timeout events at timestamp = t3 (3b).



The host server becomes operational at timestamp = t4 (4). The SLA monitor polling agent receives responses from the physical server and issues PS_reachable events at timestamp = t5 (5a). At timestamp = t6, the SLA monitoring agent captures a VM_reachable event that is generated for each virtual server (5b). The SLA management system calculates the unavailability period that affected all of the virtual servers as t6 – t2.

Pay-Per-Use Monitor

- The pay-per-use monitor mechanism measures cloud-based IT resource usage in accordance with **predefined pricing parameters and generates usage logs for fee calculations and** billing purposes.

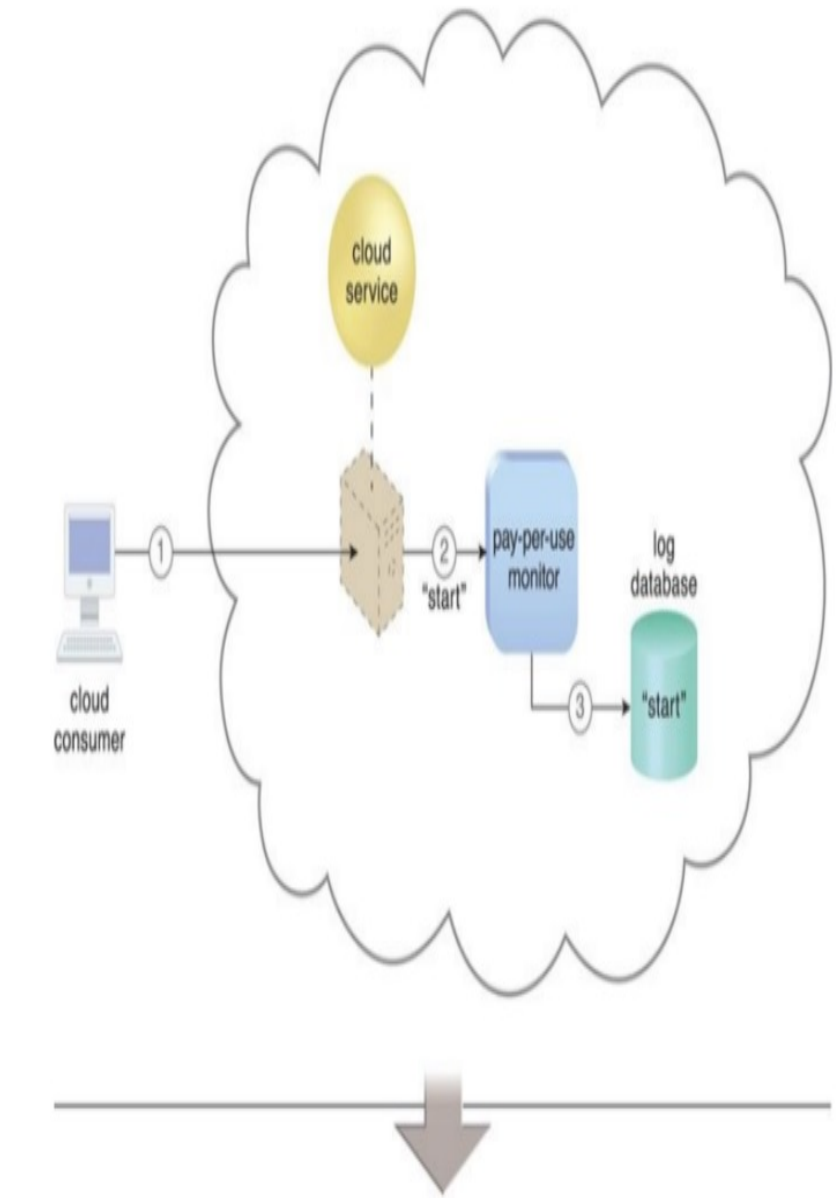
Some typical monitoring variables are:

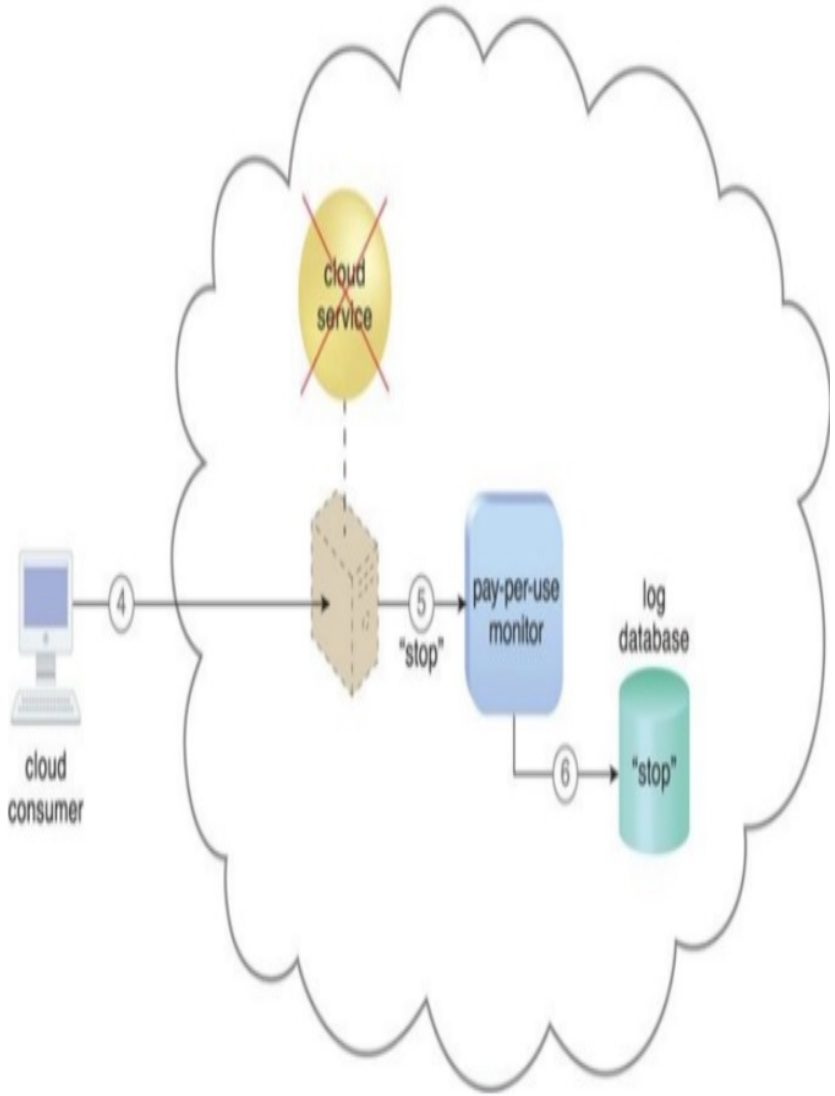
- Request/response message quantity
- Transmitted data volume
- Bandwidth consumption

The data collected by the pay-per-use monitor is processed by a billing management system that calculates the payment fees.

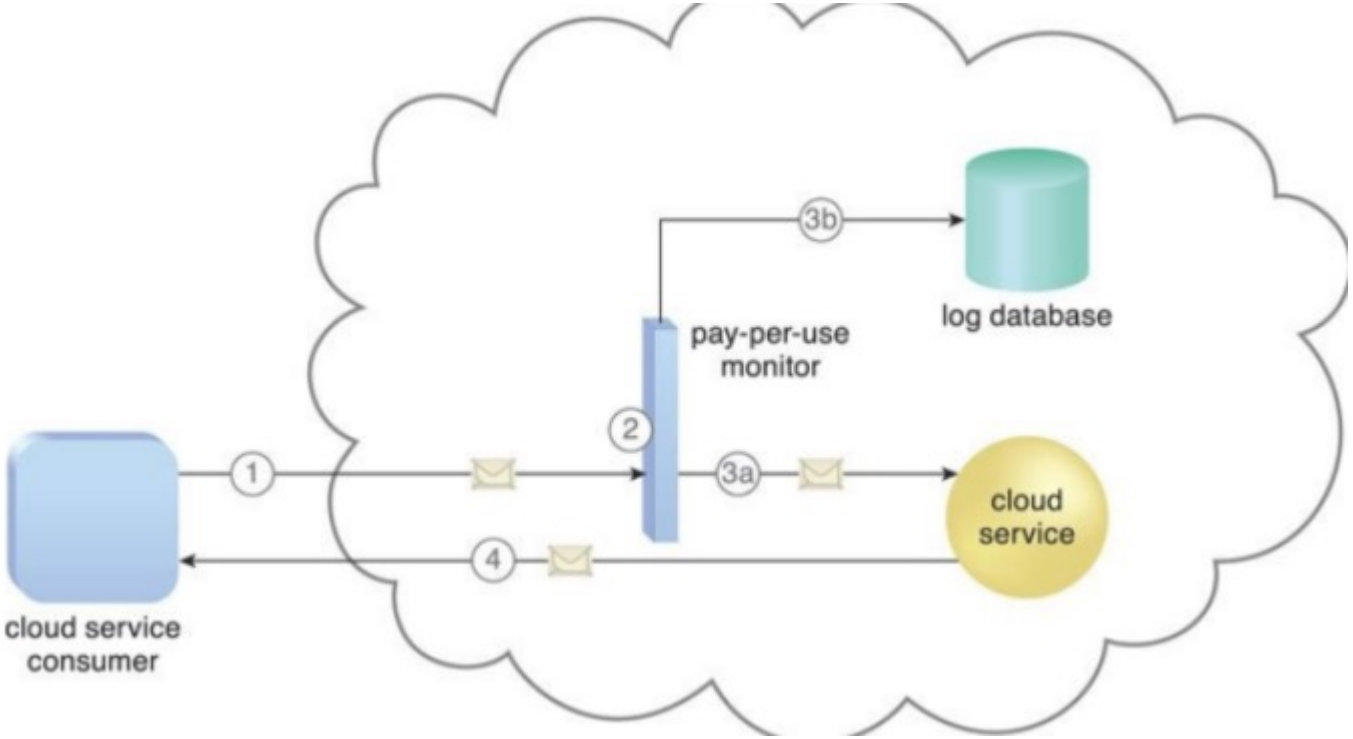
Example for pay for use monitor

A cloud consumer requests the creation of a new instance of a cloud service (1). The IT resource is instantiated and the pay-per-use monitor receives a “start” event notification from the resource software (2). The pay-per- use monitor stores the value timestamp in the log database (3). The cloud consumer later requests that the cloud service instance be stopped (4). The pay-per-use monitor receives a “stop” event notification from the resource software(5) and stores the value timestamp in the log database (6). Diagram next slide……


cloud service
cloud consumer
1
2
"start"
pay-per-use monitor
log database
3
"start"


cloud service
cloud consumer
4
5
"stop"
pay-per-use monitor
log database
6
"stop"

Pay-per-use monitor designed as a monitoring agent that transparently intercepts and analyzes runtime communication with a cloud service.



A cloud service consumer sends a request message to the cloud
service (1). The pay-per-use monitor intercepts the message (2), forwards it to the cloud service (3a), and stores the usage information in accordance with its monitoring metrics (3b). The cloud service forwards the response messages back
to the cloud service consumer to provide the requested service (4).

Failover System (Only Introduction)

- The failover system mechanism is used to increase the reliability and availability of IT resources by using established clustering technology to provide redundant implementations. A failover system is configured to automatically switch over to a redundant or standby IT resource instance whenever the currently active IT resource becomes unavailable.
- Failover systems are commonly used for mission-critical programs and reusable services that can introduce a single point of failure for multiple applications.