In this chapter we will learn

Exploring the cloud computing stack

Connecting to cloud

Understanding cloud architecture

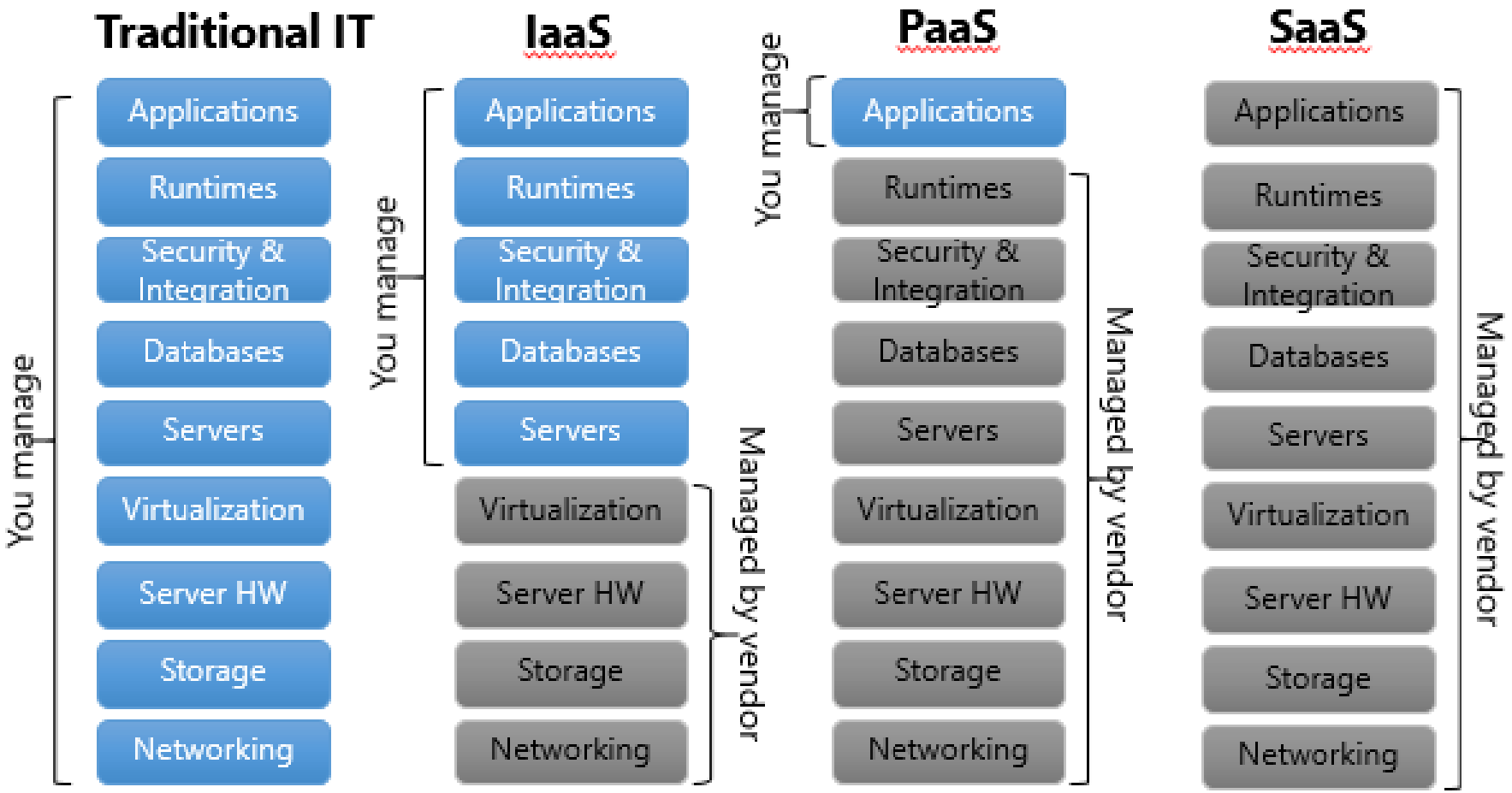
- Workload distribution architecture,
- Cloud bursting architecture,
- Disk provisioning architecture,
- Dynamic failure detection and recovery architecture

Capacity planning

Cloud mechanisms :

- Automated scaling listerner,
- Load balancer,
- Pay-per-use monitor, Audit monitor, SLA monitor,
- Fail-over Systems, Resource Cluster

Cloud computing stack



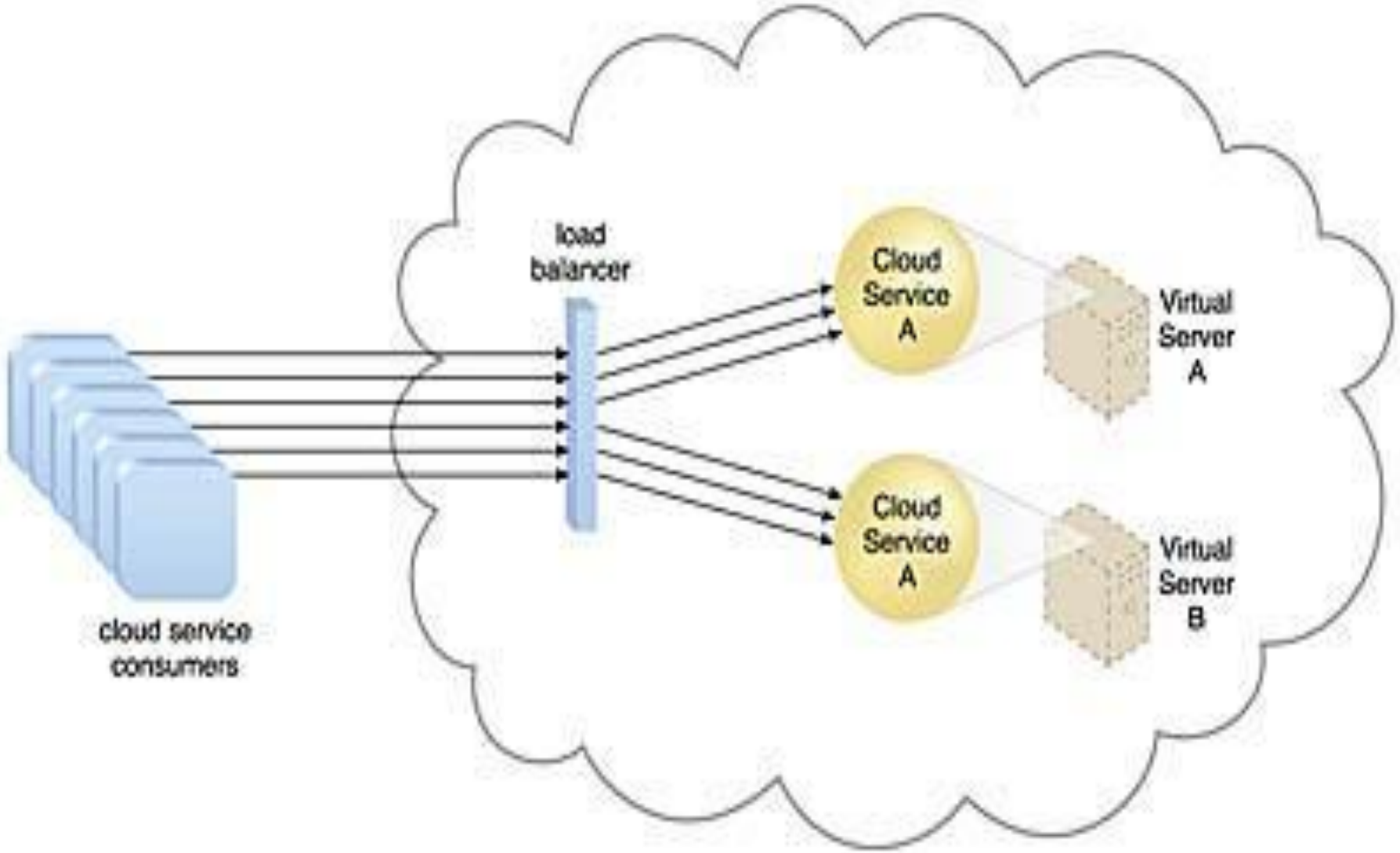
Composability	• **Composability** is a system design principle that deals with the inter-relationships of components. A highly **composable** system provides components that can be selected and assembled in various combinations to satisfy specific user requirements.
Infrastructure	• Virtual servers described in terms of a machine image or instance have characteristics that often can be described in terms of real servers delivering a certain number of microprocessor (CPU) cycles, memory access, and network bandwidth to customers.
Platforms	• Provisioning various platforms to users to customize and develop applications. Development, testing, deployments are made easier through this medium.
Virtual Appliances	• The machines that are installed in order to run services in cloud. These are platform instances in particular that can be provisioned to cloud users.
Communication Protocols	• Common XML based set of protocols used as the messaging format are the Simple Object Access Protocol (SOAP) protocol as the object model, and a set of discovery and description protocols based on the Web Services Description Language (WSDL) to manage transactions in cloud.
Applications	• Services running in the cloud

Understanding cloud architecture

What Kind of Architecture?

Workload Distribution Architecture

- Workload distribution architecture uses IT resources that can be horizontally scaled with the use of one or more identical IT resources.
- This is accomplished through the use of a load balancer that provides runtime logic which distributes the workload among the available IT assets evenly.
- This model can be applied to any IT resource and is commonly used with; distributed virtual servers, cloud storage devices, and cloud services.
- In addition to a load balancer and the previously mentioned resources, the following mechanisms can also be a part of this model:
  - Cloud Usage Monitor that can carry out run-time tracking and data processing.
  - Audit Monitor used for monitoring the system as may be required to fulfill legal requirements. Hypervisor which is used to manage workloads and virtual hosts that require distribution. Logical network perimeter which isolates cloud consumer network boundaries. Resource clusters commonly used to support workload balancing between cluster nodes. Resource replication which generates new instances of virtualized resources under increased workloads.


load
balancer
Cloud
Service
A
Cloud
Service
A
Virtual
Server
A
Virtual
Server
B
cloud service
consumers

Working of Workload Distribution Architecture:-

The workload architecture model basically functions as follows:

- Resource A and resource B are exact copies of the same resource.
- Inbound requests from consumers are handled by the load balancer which forwards the request to the appropriate resource dependent on workload being handled by each resource.
- In other words, if resource A is busier than resource B, it will forward the resource request to resource B.
- In this manner this model distributes the load among the available IT resources based on workload of each resource.

In addition to the base load balancer mechanism, and the virtual server and cloud storage device mechanisms to which load balancing can be applied, the following mechanisms can also be part of this cloud architecture:

Audit Monitor – When distributing runtime workloads, the type and geographical location of the IT resources that process the data can determine whether monitoring is necessary to fulfill legal and regulatory requirements.

Cloud Usage Monitor – Various monitors can be involved to carry out runtime workload tracking and data processing.

Hypervisor – Workloads between hypervisors and the virtual servers that they host may require distribution.

Logical Network Perimeter – The logical network perimeter isolates cloud consumer network boundaries in relation to how and where workloads are distributed.

Resource Cluster – Clustered IT resources in-active/active mode are commonly used to support workload balancing between different cluster nodes.

Resource Replication – This mechanism can generate new instances of virtualized IT resources in response to runtime workload distribution demands

Cloud Bursting architecture:-

Cloud Bursting

Cloud bursting is an application deployment model in which an application runs in an internal cloud or a proprietary computing architecture which provides hosted services to a limited number of people (also known as a private cloud).

This kind of architecture is behind a company's firewall. The application could also be at a data centre.

When the demand of the computing processes reaches its capacity and begins to spike, it **"bursts"** into a public cloud (available to the general public) when the demand for computing capacity sharply increases.

Capacity planning

- For available resources, capacity planning seeks a heavy demand. It determines whether the systems are working properly, used to measure their performance, determine the usage of patterns and predict future demand of cloud-capacity. This also adds an expertise planning for improvement and optimizes performance. The goal of capacity planning is to maintain the workload without improving the efficiency. Tuning of performance and work optimization is not the major target of capacity planners.
- It measures the maximum amount of task that it can perform. The capacity planning for cloud technology offers the systems with more enhanced capabilities including some new challenges over a purely physical system.

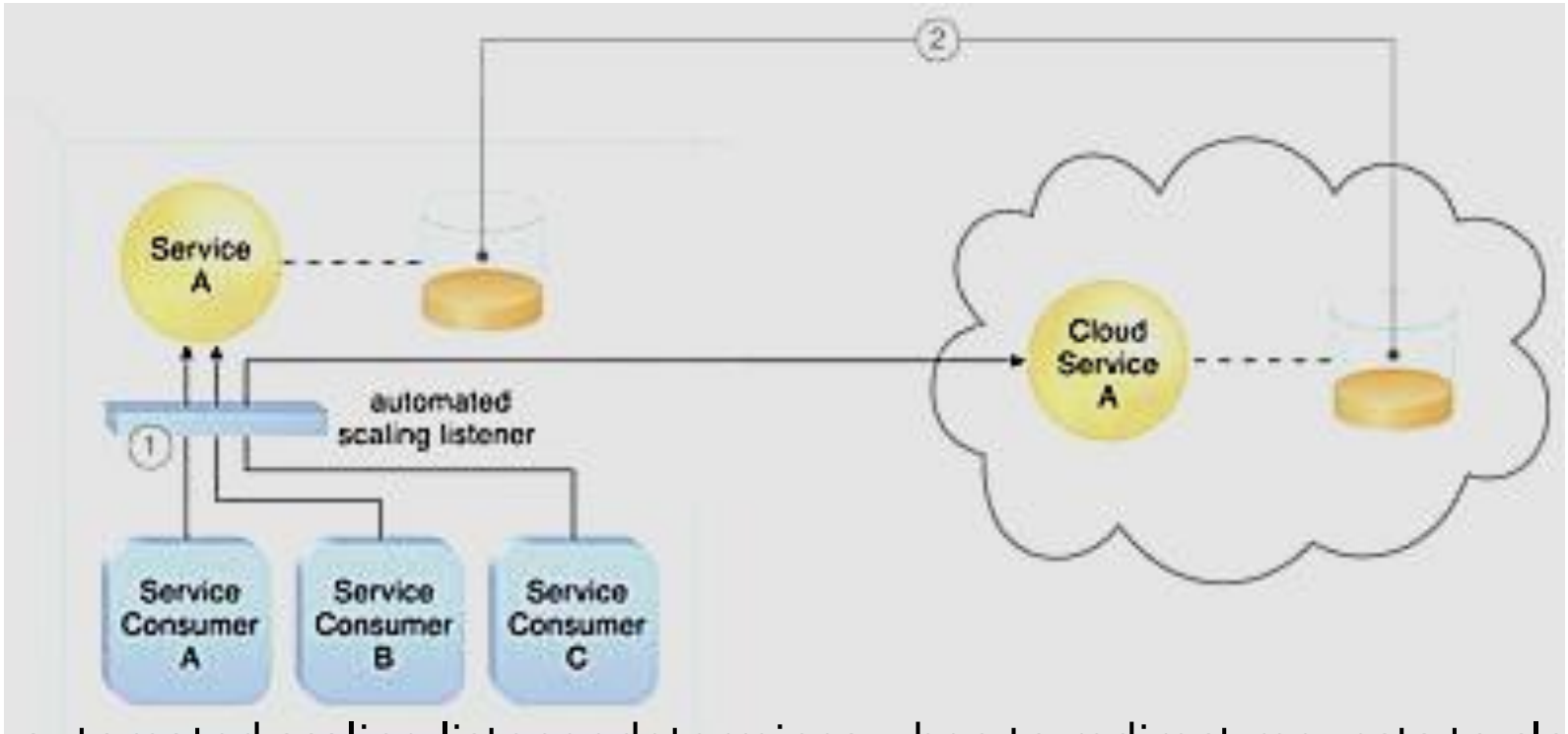
Steps in Capacity planning

- Determine the distinctiveness of the present system.
- Determine the working load for different resources in the system such as CPU, RAM, network, etc.
- Load the system until it gets overloaded; & state what's requiring to uphold acceptable performance.
- Predict the future based on older statistical reports & other factors.
- Deploy resources to meet the predictions & calculations.
- Repeat step (i) through (v) as a loop.

Cloud bursting architecture

The *cloud bursting architecture* establishes a form of dynamic scaling that scales or "bursts out" on-premise IT resources into a cloud whenever predefined capacity thresholds have been reached. The corresponding cloud-based IT resources are redundantly pre-deployed but remain inactive until cloud bursting occurs. After they are no longer required, the cloud-based IT resources are released and the architecture "bursts in" back to the on-premise environment.

Cloud bursting is a flexible scaling architecture that provides cloud consumers with the option of using cloud-based IT resources only to meet higher usage demands. The foundation of this architectural model is based on the automated scaling listener and resource replication mechanisms.



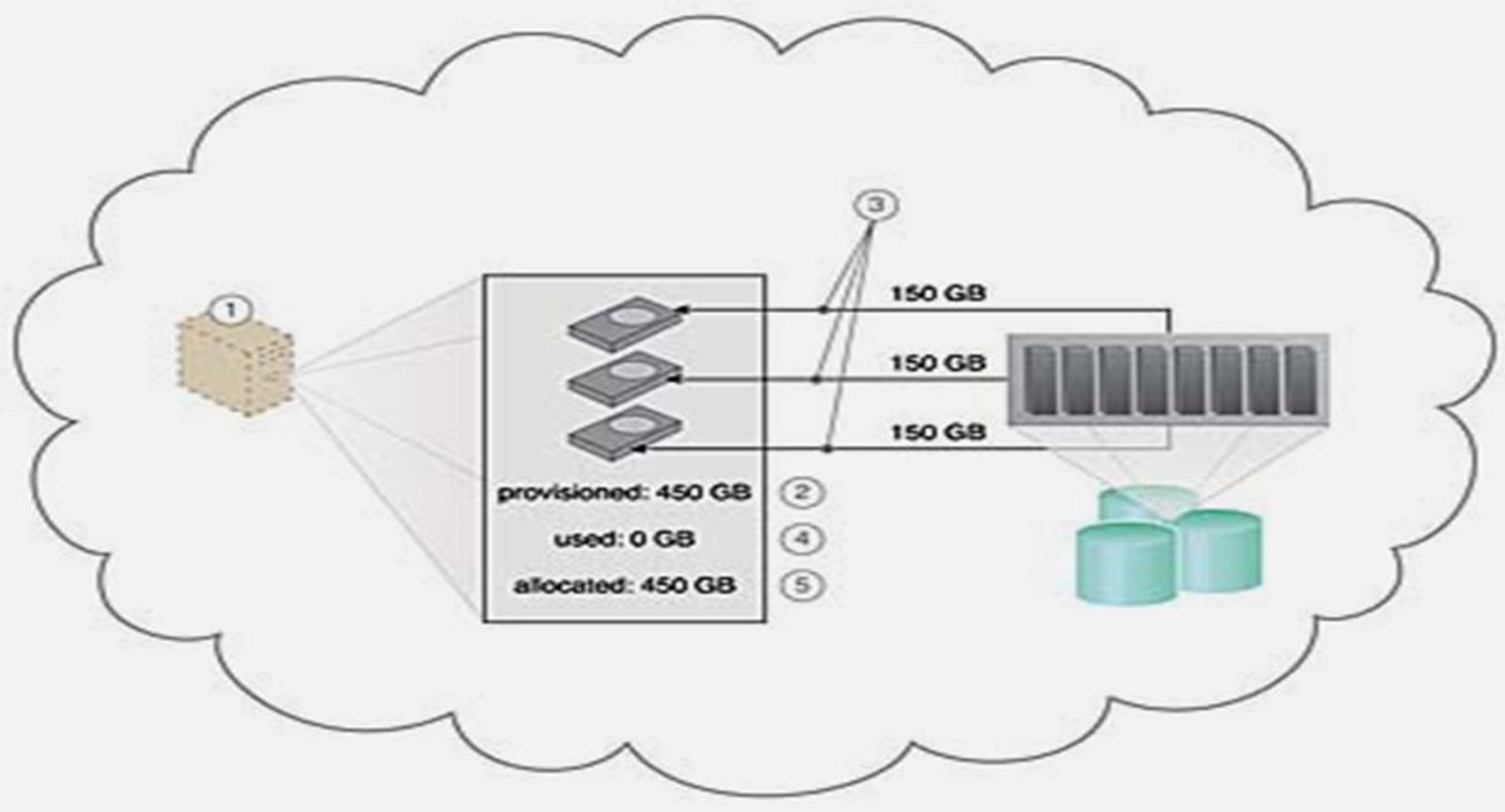
The automated scaling listener determines when to redirect requests to cloud-based IT resources, and resource replication is used to maintain synchronicity between on-premise and cloud-based IT resources in relation to state information

An automated scaling listener monitors the usage of on-premise Service A, and redirects Service Consumer C's request to Service A's redundant implementation in the cloud (Cloud Service A) once Service A's usage threshold has been exceeded (1). A resource replication system is used to keep state management databases synchronized (2).

Elastic Disk Provisioning Architecture

Cloud consumers are commonly charged for cloud-based storage space based on fixed-disk storage allocation, meaning the charges are predetermined by disk capacity and not aligned with actual data storage consumption. Figure demonstrates this by illustrating a scenario in which a cloud consumer provisions a virtual server with the Windows Server operating system and three 150 GB hard drives. The cloud consumer is billed for using 450 GB of storage space after installing the operating system, even though the operating system only requires 15 GB of storage space.

Elastic Disk Provisioning Architecture

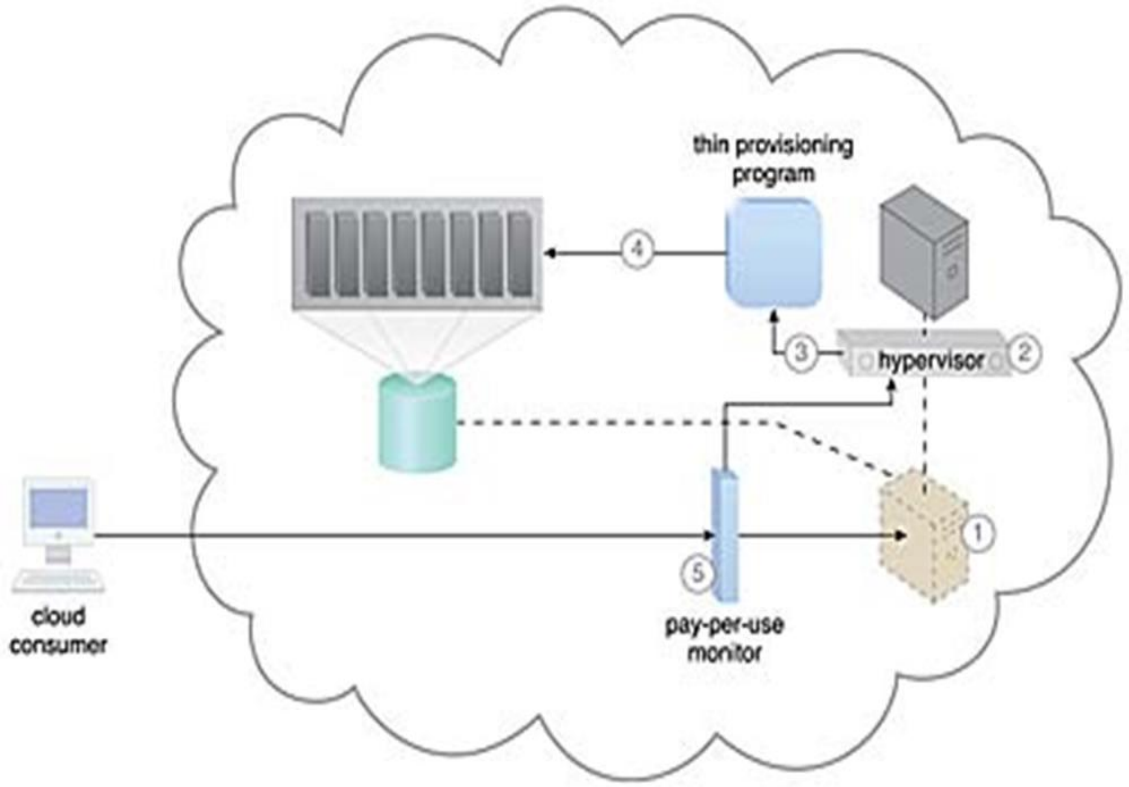


Elastic Disk Provisioning Architecture

The elastic disk provisioning architecture establishes a dynamic storage provisioning system that ensures that the cloud consumer is granularly billed for the exact amount of storage that it actually uses. This system uses thin-provisioning technology for the dynamic allocation of storage space, and is further supported by runtime usage monitoring to collect accurate usage data for billing purposes.

Thin-provisioning software is installed on virtual servers that process dynamic storage allocation via the hypervisor, while the pay-per-use monitor tracks and reports granular billing-related disk usage data.

Elastic Disk Provisioning Architecture



A request is received from a cloud consumer, and the provisioning of a new virtual server instance begins

(1) As part of the provisioning process, the hard disks are chosen as dynamic or thin-provisioned disks.

(2) The hypervisor calls a dynamic disk allocation component to create thin disks for the virtual server.

(3) Virtual server disks are created via the thin-provisioning program and saved in a folder of near-zero size.

(4)The size of this folder and its files grow as operating applications are installed and additional files are copied onto the virtual server.

(5) The pay-per-use monitor tracks the actual dynamically allocated storage for billing purposes.

The following mechanisms can be included in this architecture in addition to the cloud storage device, virtual server, hypervisor, and pay-per-use monitor:

Cloud Usage Monitor – Specialized cloud usage monitors can be used to track and log storage usage fluctuations.

Resource Replication – Resource replication is part of an elastic disk provisioning system when conversion of dynamic thin-disk storage into static thick-disk storage is required.

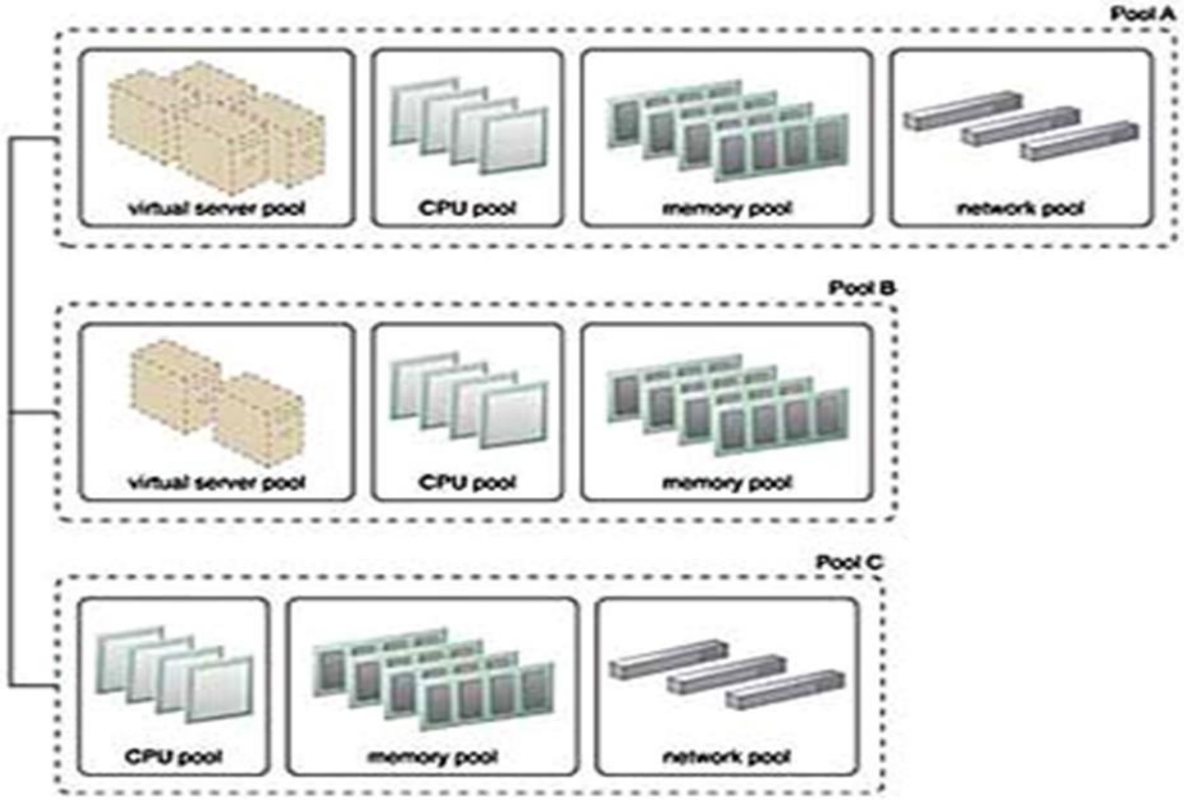
Resource Pooling Architecture

A resource pooling architecture is based on the use of one or more resource pools, in which identical IT resources are grouped and maintained by a system that automatically ensures that they remain synchronized.

- Physical server pools are composed of networked servers that have been installed with operating systems and other necessary programs and/or applications and are ready for immediate use.
- Virtual server pools are usually configured using one of several available templates chosen by the cloud consumer during provisioning. For example, a cloud consumer can set up a pool of mid-tier Windows servers with 4 GB of RAM or a pool of low-tier Ubuntu servers with 2 GB of RAM.
- Storage pools, or cloud storage device pools, consist of file-based or block-based storage structures that contain empty and/or filled cloud storage devices.
- Network pools (or interconnect pools) are composed of different preconfigured network connectivity devices. For example, a pool of virtual firewall devices or physical network switches can be created for redundant connectivity, load balancing, or link aggregation.
- CPU pools are ready to be allocated to virtual servers, and are typically broken down into individual processing cores.
- Pools of physical RAM can be used in newly provisioned physical servers or to vertically scale physical servers.

Resource Pooling Architecture

Dedicated pools can be created for each type of IT resource and individual pools can be grouped into a larger pool, in which case each individual pool becomes a sub-pool. An example of simple resource pool can bee seen below.

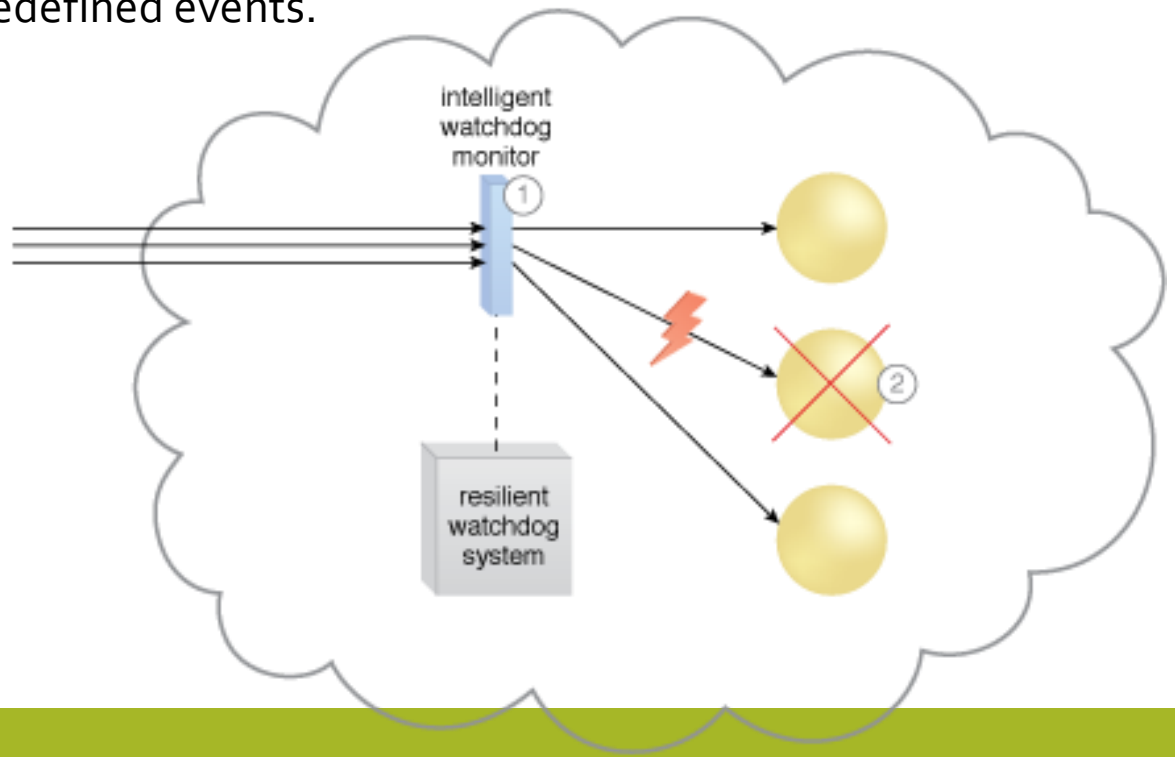


Resource pools can become highly complex, with multiple pools created for specific cloud consumers or applications. A hierarchical structure can be established to form parent, sibling, and nested pools in order to facilitate the organization of diverse resource pooling requirements.

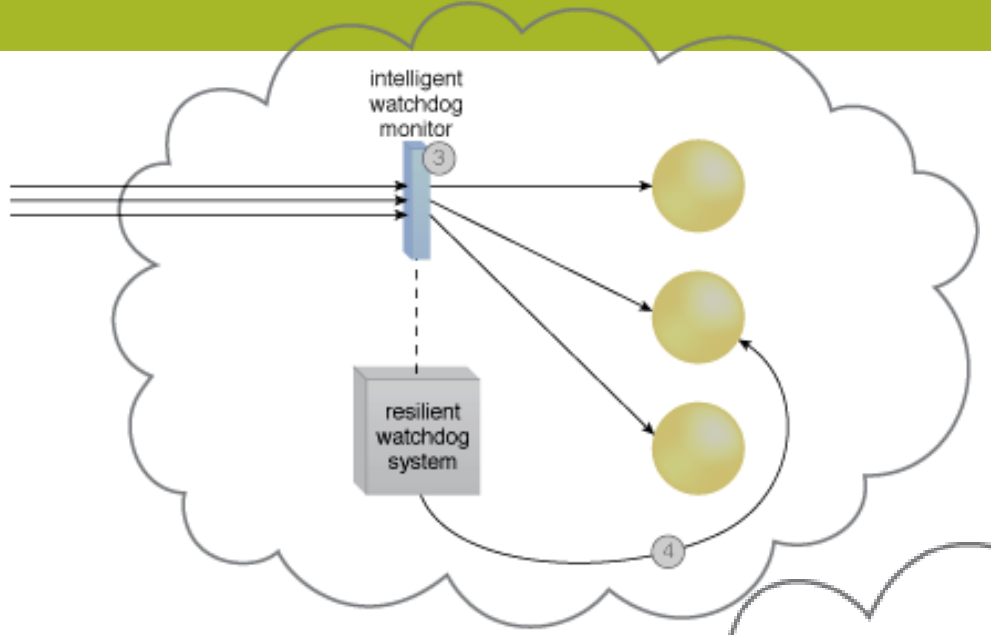
Dynamic Failure Detection and Recovery Architecture

Cloud-based environments can be comprised of vast quantities of IT resources that are simultaneously accessed by numerous cloud consumers. Any of those IT resources can experience failure conditions that require more than manual intervention to resolve. Manually administering and solving IT resource failures is generally inefficient.

The dynamic failure detection and recovery architecture establishes a resilient watchdog system to monitor and respond to a wide range of pre-defined failure scenarios. This system notifies and escalates the failure conditions that it cannot automatically resolve itself. It relies on specialized cloud storage usage monitor called the intelligent watchdog monitor to actively track IT resources and take pre-defined tasks and actions to predefined events.

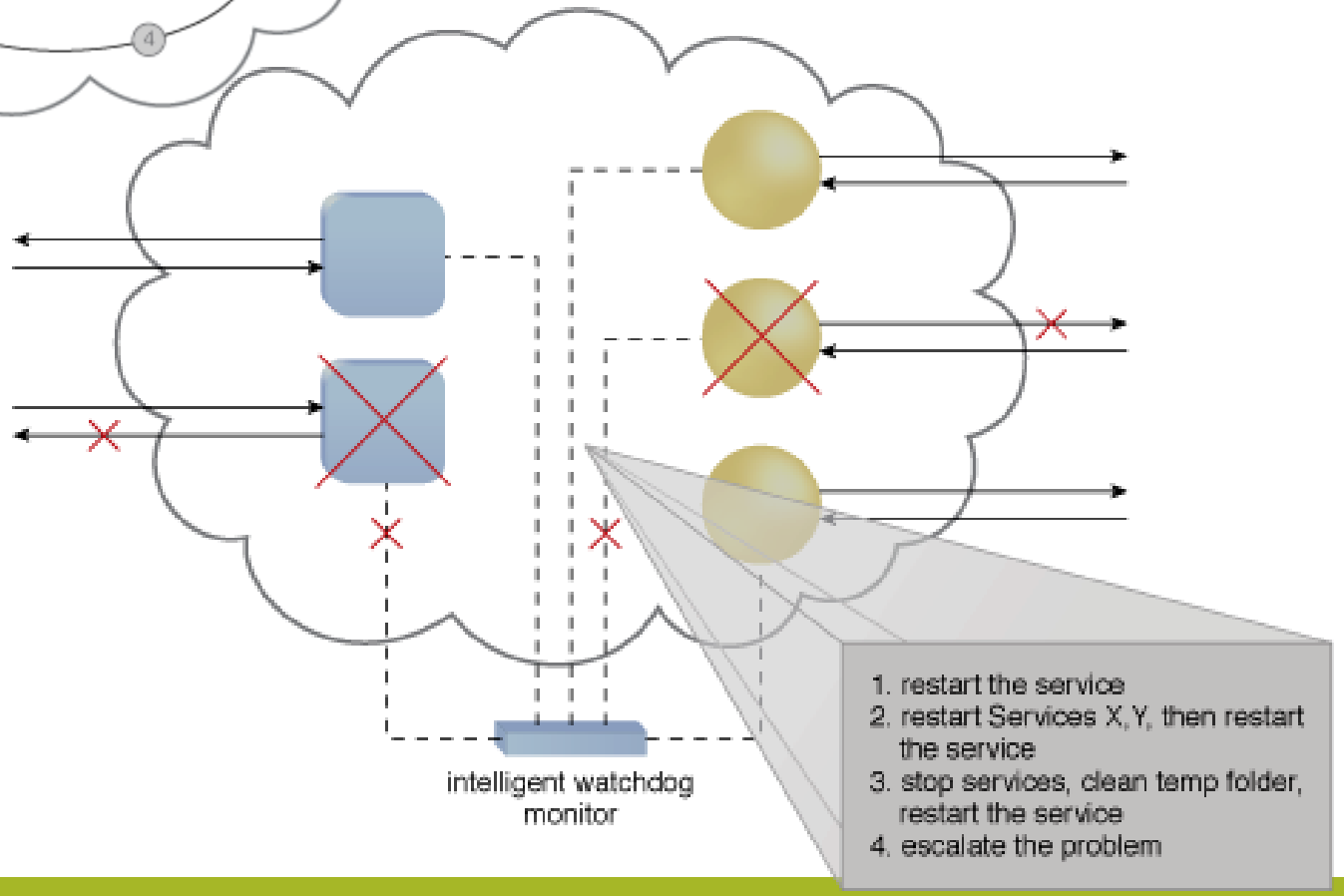


(1)The intelligent watchdog monitor keeps track of cloud consumer requests,
(2) and detects that a cloud service has failed.



The intelligent watchdog monitor notifies the resilient watchdog system (3), which restores the cloud service based on predefined policies (4).

In the event of any failures, the active monitor refers to its predefined policies to recover the service step by step, escalating the processes as the problem proves to be deeper than expected.



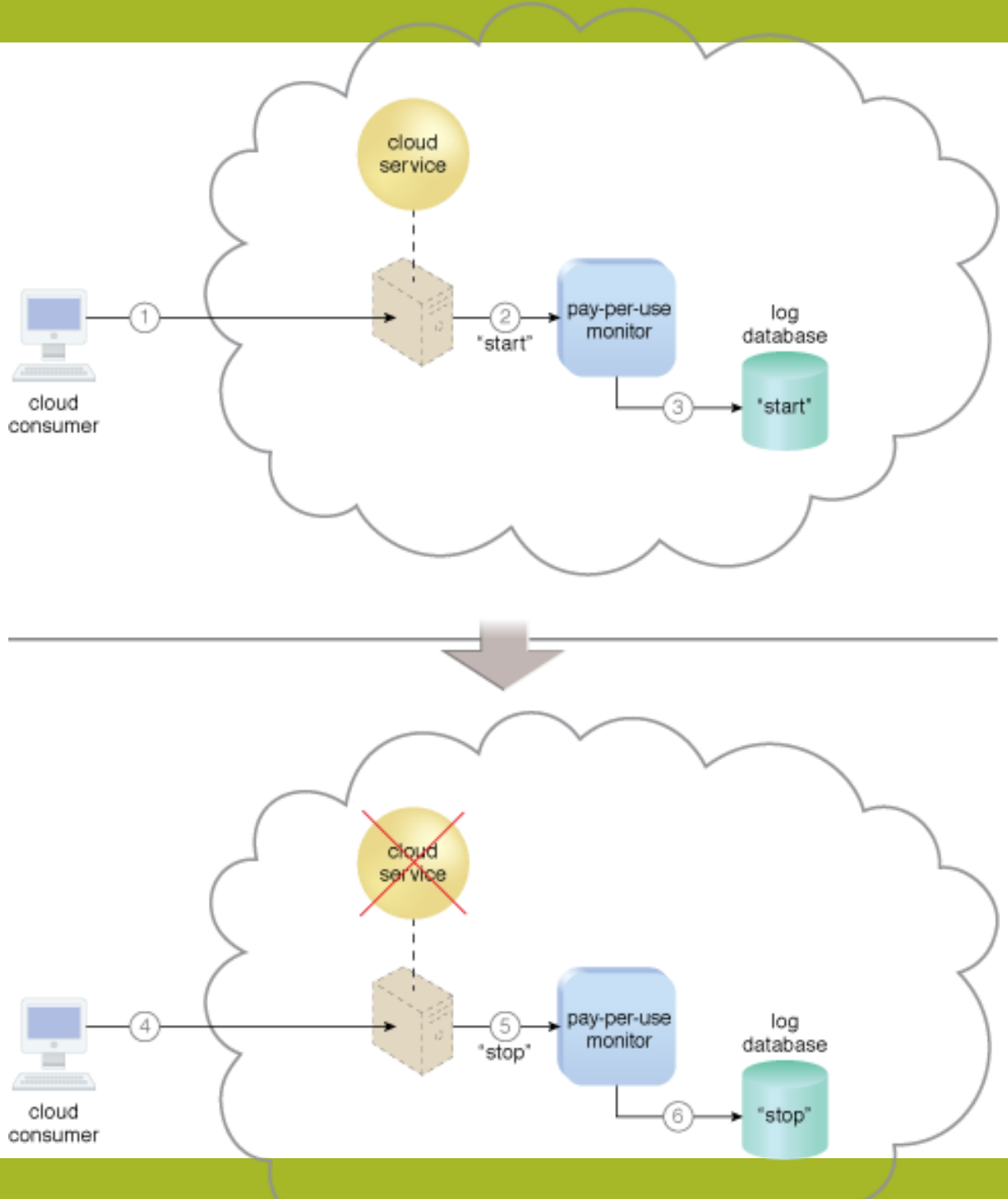
Pay-Per-Use Monitor

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

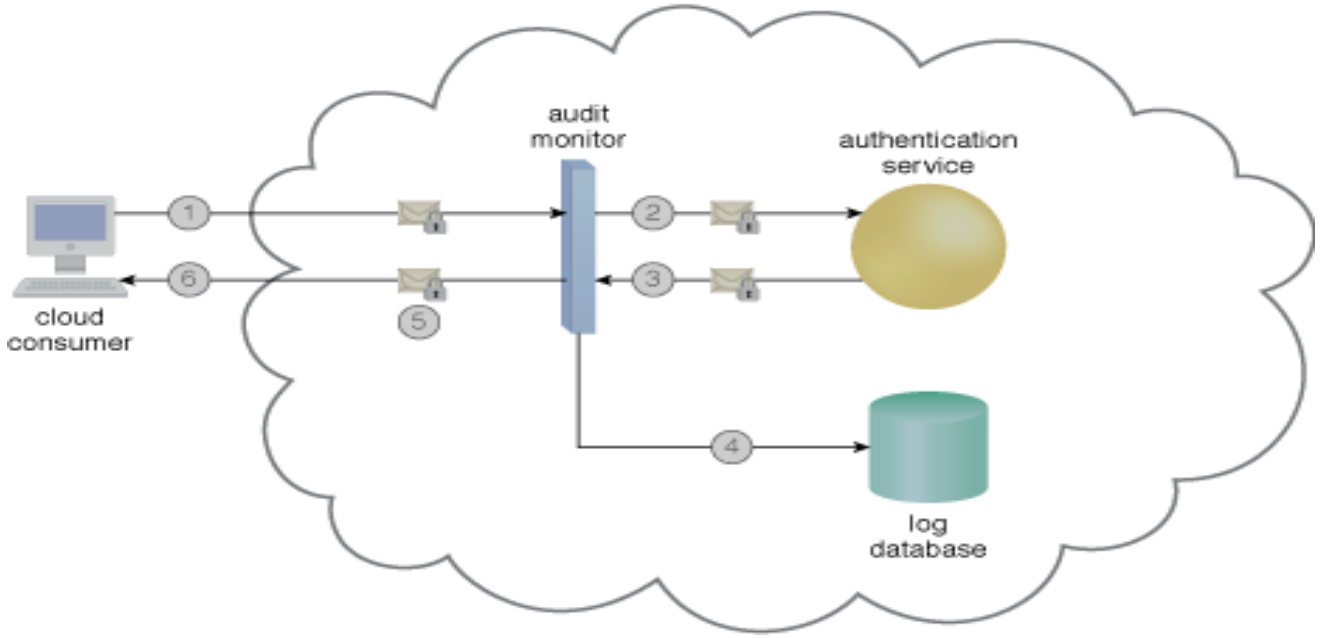
The data collected by the pay-per-use monitor is processed by a billing management system that calculates the payment fees.



1) A cloud consumer requests the creation of a new instance of a cloud service
2) The IT resource is instantiated and they pay-per-use monitor mechanism receives a "start" event notification from the resource software
3) The pay-per-use monitor stores the value timestamp in the log database
4) The cloud consumer later requests that the cloud service instance be stopped
5) The pay-per-use monitor receives a "stop" event notification from the resource software and stores the value timestamp in the log database

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. The figure 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.

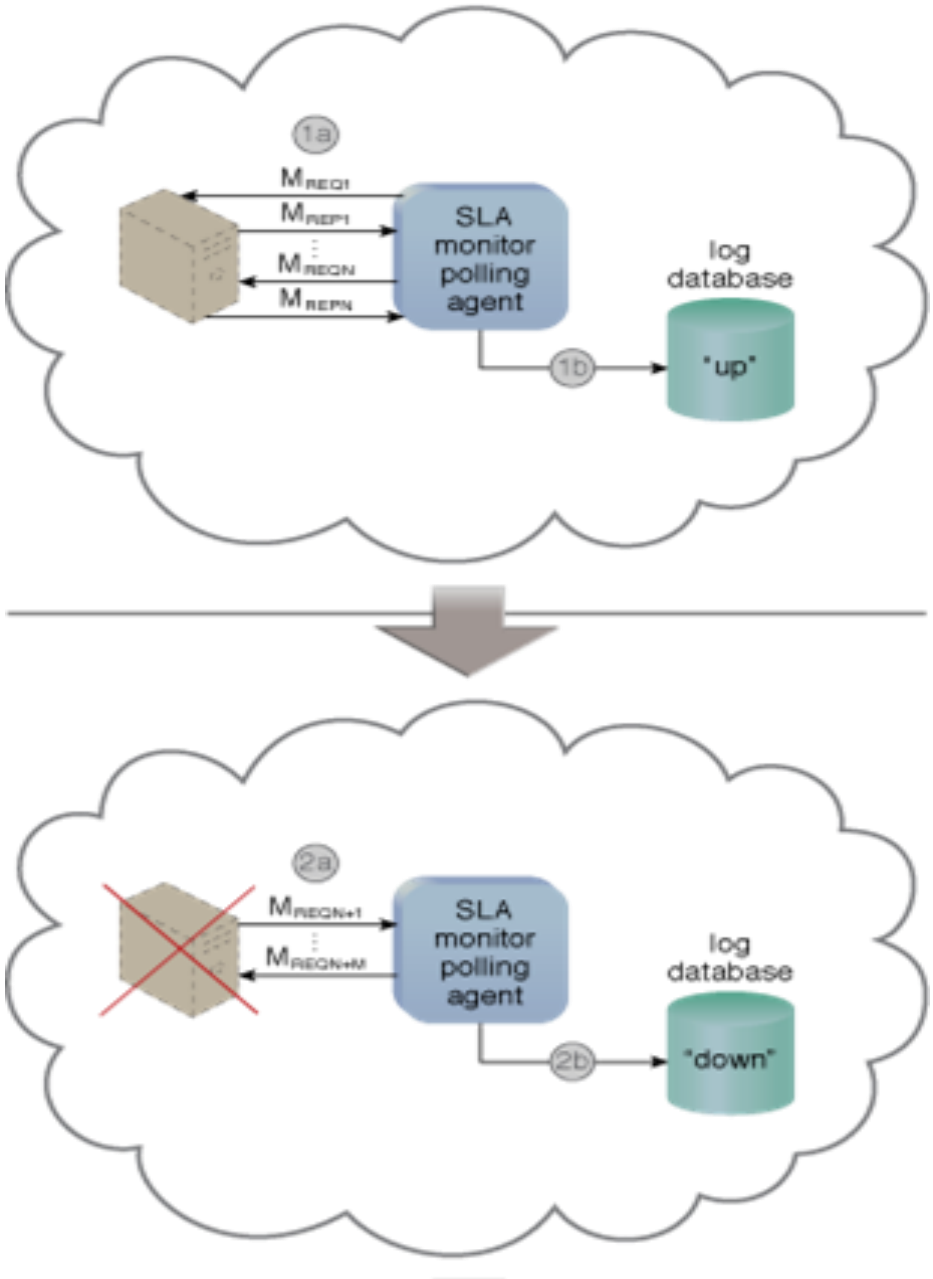


1) A cloud service consumer requests access to a cloud service by sending a login request message with security credentials.
2) The audit monitor intercepts the message
3) And forwards it to the authentication service.
4) The authentication service processes the security credentials. A response message is generated for the cloud service consumer, in addition to the results from the login attempt.
5) 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.
6) Access has been granted, and a response is sent back to the cloud service consumer.

SLA Management System

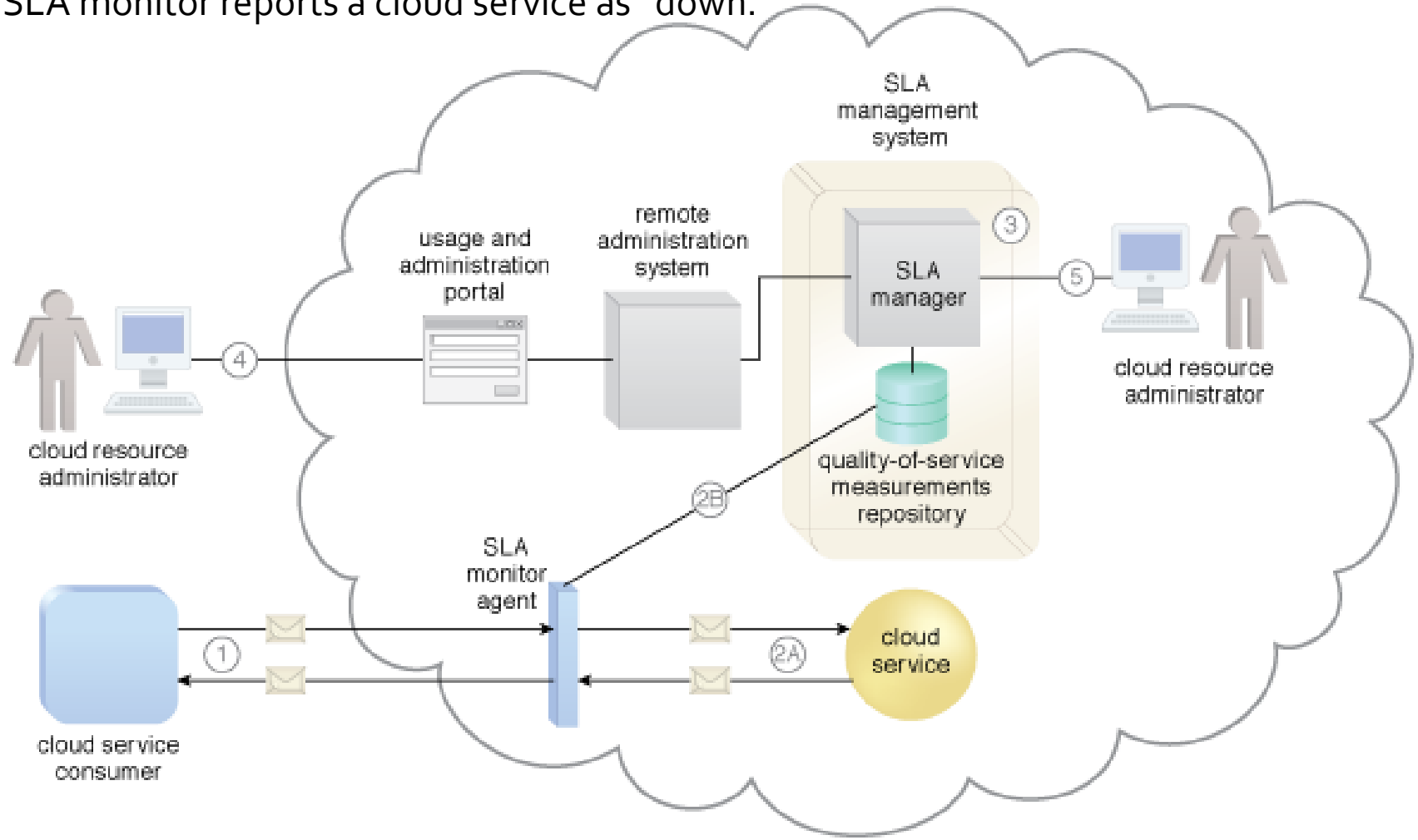
The SLA management system mechanism represents a range of commercially available cloud management products that provide features pertaining to the administration, collection, storage, reporting, and runtime notification of SLA data.

An SLA management system deployment will generally include a repository used to store and retrieve collected SLA data based on pre-defined metrics and reporting parameters. It will further rely on one or more SLA monitor mechanisms to collect the SLA data that can then be made available in near-realtime to usage and administration portals to provide ongoing feedback regarding active cloud services (Figure 1). The metrics monitored for individual cloud services are aligned with the SLA guarantees in corresponding cloud provisioning contracts.



A cloud service consumer interacts with a cloud service (1).
An SLA monitor intercepts the exchanged messages, evaluates the interaction, and collects relevant runtime data in relation to quality-of-service guarantees defined in the cloud service's SLA (2A).
The data collected is stored in a repository (2B) that is part of the SLA management system (3).
Queries can be issued and reports can be generated for an external cloud resource administrator via a usage and administration portal (4) or for an internal cloud resource administrator via the SLA management system's native user-interface (5).

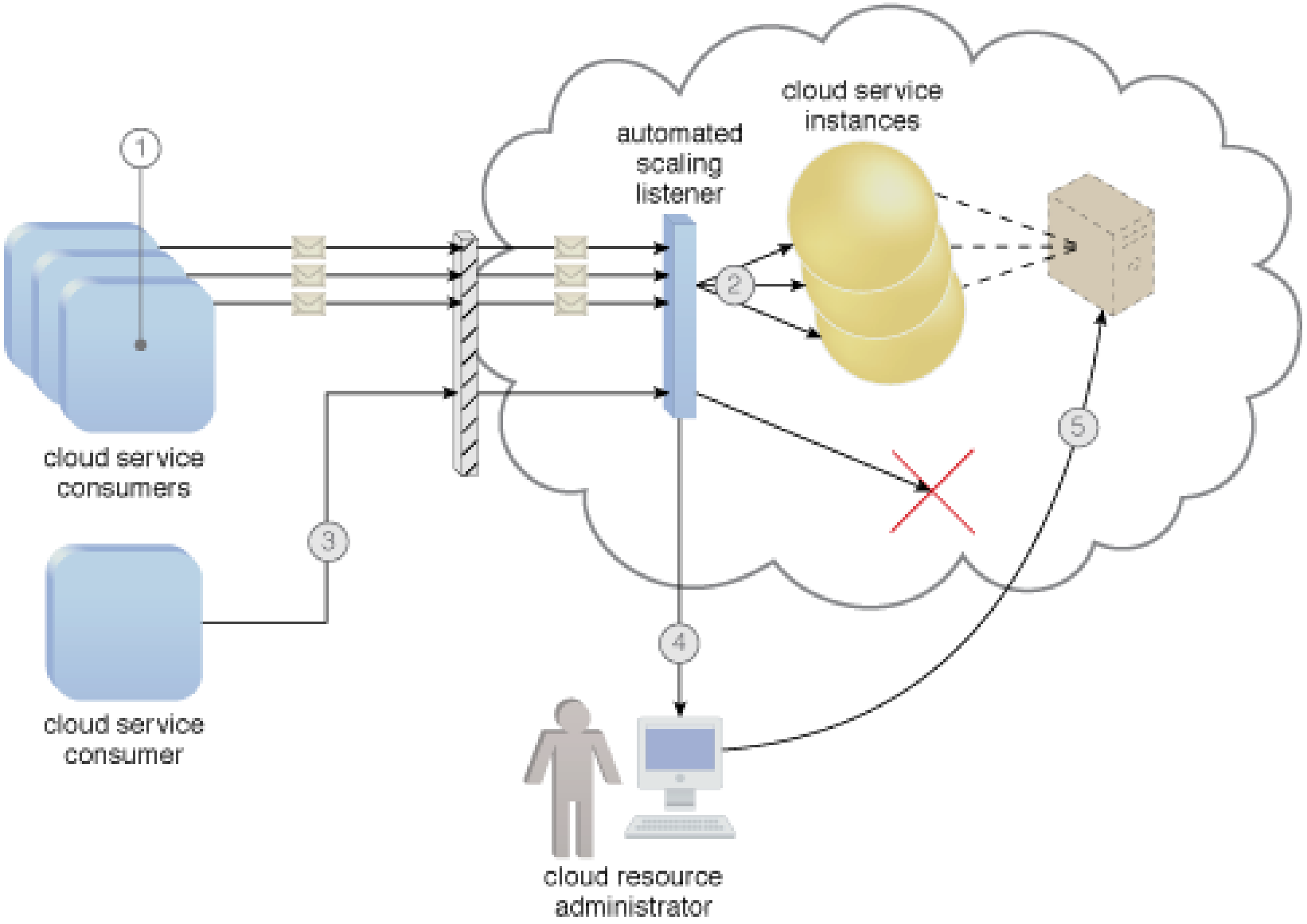
The SLA monitor mechanism is used to specifically observe the runtime performance of cloud services to ensure that they are fulfilling the contractual QoS requirements published in SLAs (Figure 1). The data collected by the SLA monitor is processed by an SLA management system to be aggregated into SLA reporting metrics. This system can proactively repair or failover cloud services when exception conditions occur, such as when the SLA monitor reports a cloud service as "down."



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.

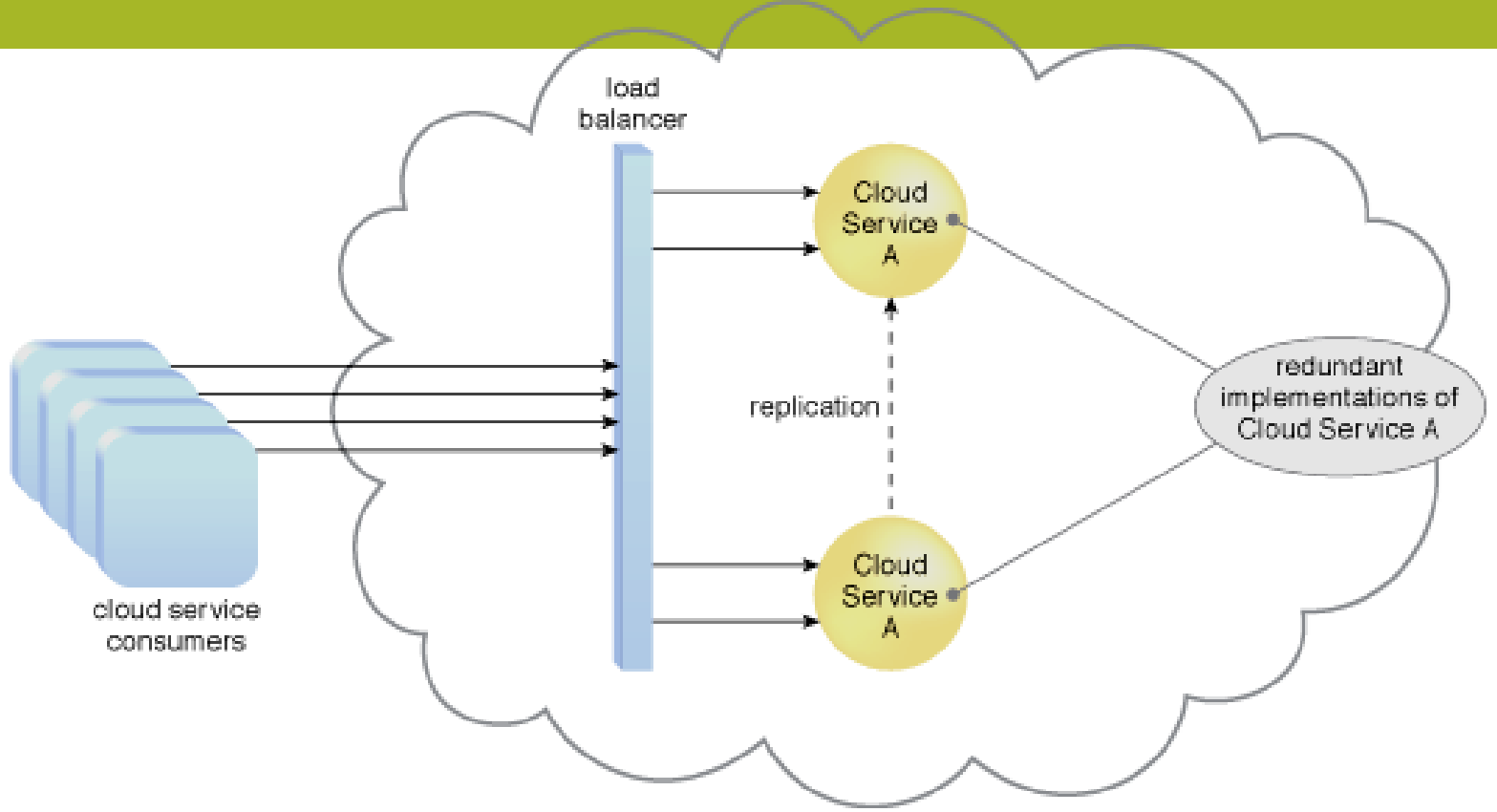
- Automated scaling listeners can provide different types of responses to workload fluctuation conditions, such as:
- Automatically scaling IT resources out or in based on parameters previously defined by the cloud consumer (commonly referred to as auto-scaling).
- Automatic notification of the cloud consumer when workloads exceed current thresholds or fall below allocated resources. This way, the cloud consumer can choose to adjust its current IT resource allocation.
- Different cloud provider vendors have different names for service agents that act as automated scaling listeners.


cloud service instances
automated scaling listener
1
2
3
4
5
cloud service consumers
cloud service consumer
cloud resource administrator

Load Balancer

The load balancer mechanism is a runtime agent with logic fundamentally based on the premise of employing horizontal scaling to balance a workload across two or more IT resources to increase performance and capacity beyond what a single IT resource can provide. Beyond simple division of labor algorithms, 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 clouds 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.

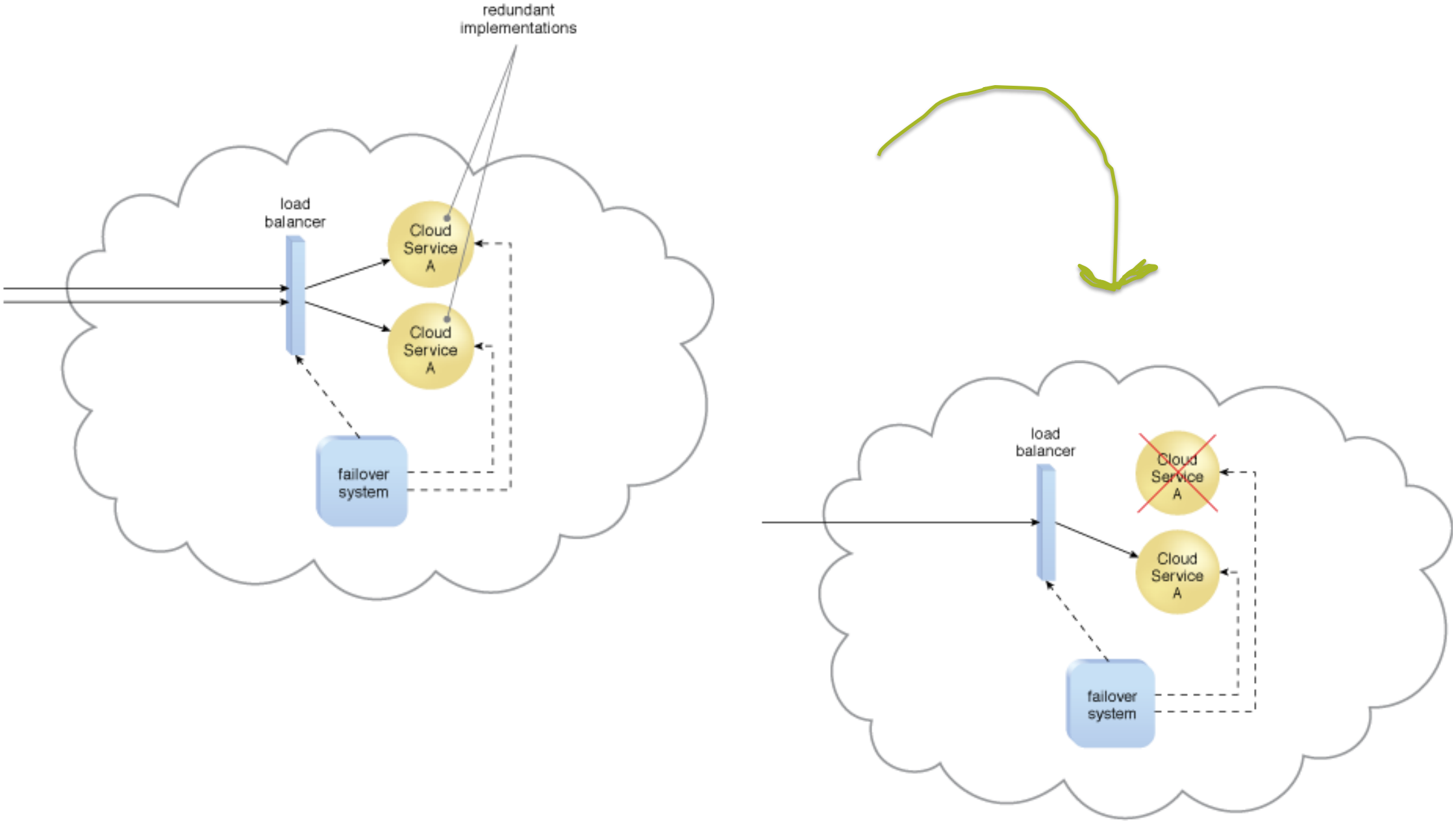
The load balancer mechanisms can exist as a:

- 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.

Fail over systems

- 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.
- A failover system can span more than one geographical region so that each location hosts one or more redundant implementations of the same IT resource.


redundant
implementations
load
balancer
Cloud
Service
A
Cloud
Service
A
failover
system
load
balancer
Cloud
Service
A
Cloud
Service
A
failover
system

Service level agreements

A Service Level Agreement (SLA) is the bond for performance negotiated between the cloud services provider and the client. Earlier, in cloud computing all Service Level Agreements were negotiated between a client and the service consumer. Nowadays, with the initiation of large utility-like cloud computing providers, most Service Level Agreements are standardized until a client becomes a large consumer of cloud services. Service level agreements are also defined at different levels which are mentioned below:

- Customer-based SLA
- Service-based SLA
- Multi-level SLA

Few Service Level Agreements are enforceable as contracts, but mostly are agreements or contracts which are more along the lines of an Operating Level Agreement (OLA) and may not have the restriction of law. It is fine to have an attorney review the documents before making a major agreement to the cloud service provider. Service Level Agreements usually specify some parameters which are mentioned below:

- Availability of the Service (uptime)
- Latency or the response time
- Service components reliability
- Each party accountability
- Warranties

In any case, if a cloud service provider fails to meet the stated targets of minimums then the provider has to pay the penalty to the cloud service consumer as per the agreement. So, Service Level Agreements are like insurance policies in which the corporation has to pay as per the agreements if any casualty occurs.

Service Oriented Architecture

- SOA, or service-oriented architecture, defines a way to make software components reusable via service interfaces. These interfaces utilize common communication standards in such a way that they can be rapidly incorporated into new applications without having to perform deep integration each time.

- Each service in an SOA embodies the code and data integrations required to execute a complete, discrete business function (e.g., checking a customer's credit, calculating a monthly loan payment, or processing a mortgage application). The service interfaces provide loose coupling, meaning they can be called with little or no knowledge of how the integration is implemented underneath. The services are exposed using standard network protocols—such as SOAP (simple object access protocol)/HTTP or JSON/HTTP—to send requests to read or change data. The services are published in a way that enables developers to quickly find them and reuse them to assemble new applications.

Benefits of SOA

- Greater business agility; faster time to market: The efficiency of assembling applications from reusable service interfaces, rather than rewriting and reintegrating with every new development project, enables developers to build applications much more quickly in response to new business opportunities.
- Ability to leverage legacy functionality in new markets: A well-crafted SOA enables developers to easily take functionality ‘locked’ in one computing platform or environment and extend it to new environments and markets. For example, many companies have used SOA to expose functionality from mainframe-based financial systems to the web, enabling their customers to serve themselves to processes and information previously accessible only through direct interaction with the company’s employees or business partners.
- Improved collaboration between business and IT: In an SOA, services can be defined in business terms (e.g., ‘generate insurance quote’ or ‘calculate capital equipment ROI’). This enables business analysts to work more effectively with developers on important insights—such as the scope of a business process defined by a service or the business implications of changing a process—that can lead to a better result.