All Questions Index Page

1. Discuss the concept of Network Centric Computing and Network Centric content. Identify their characteristics and advantages.

2. Describe briefly with an architectural diagram the services offered by AWS, accessible from the AWS Management Console.

3. Classify cloud computing delivery models based on the level of consumer control with neat diagrams.

4. Describe with a diagram the architecture of Windows Azure.

5. Describe using diagrams the basic workflow patterns.

6. illustrate with a diagram how the Cirrus platform helps in executing legacy Windows applications on the cloud.

7. What is the concept of a task in workflow modeling? Identify the attributes of a task and describe different types of tasks.

8. Describe the steps in the workflow of the GrepTheWeb application with a diagram.

9. Draw the NIST cloud computing reference model and discuss the function of the entities involved.

10. Describe and compare the three storage services offered by AWS.

11. Illustrate with a diagram, responsibility sharing between the user and cloud service provider.

12. Illustrate with necessary diagrams the RAID-5 technology, and show with an example how it can be used to provide reliable data storage on the cloud.

13. Justify with diagrams the parallel between workflows and programs.

14. Describe the map-reduce philosophy with a neat diagram and an Example.

15. Describe with necessary diagrams the organization of the Zookeeper service and the processing of read and write operations.

16. Illustrate with a diagram how the BigJob software system can be used for the execution of loosely coupled workloads on the Azure platform.

1. a) Discuss the concept of Network Centric Computing and Network Centric content. Identify their characteristics and advantages.

Network Centric Computing

Information processing can be done more efficiently on large farms of computing and storage systems accessible via the Internet.
Grid computing – initiated by the National Labs in the early 1990s; targeted primarily at scientific computing.
Utility computing – initiated in 2005-2006 by IT companies and targeted at enterprise computing.

The focus of utility computing is on the business model for providing computing services; it often requires a cloud-like infrastructure.
Cloud computing is a path to utility computing embraced by major IT companies including: Amazon, HP, IBM, Microsoft, Oracle, and others.

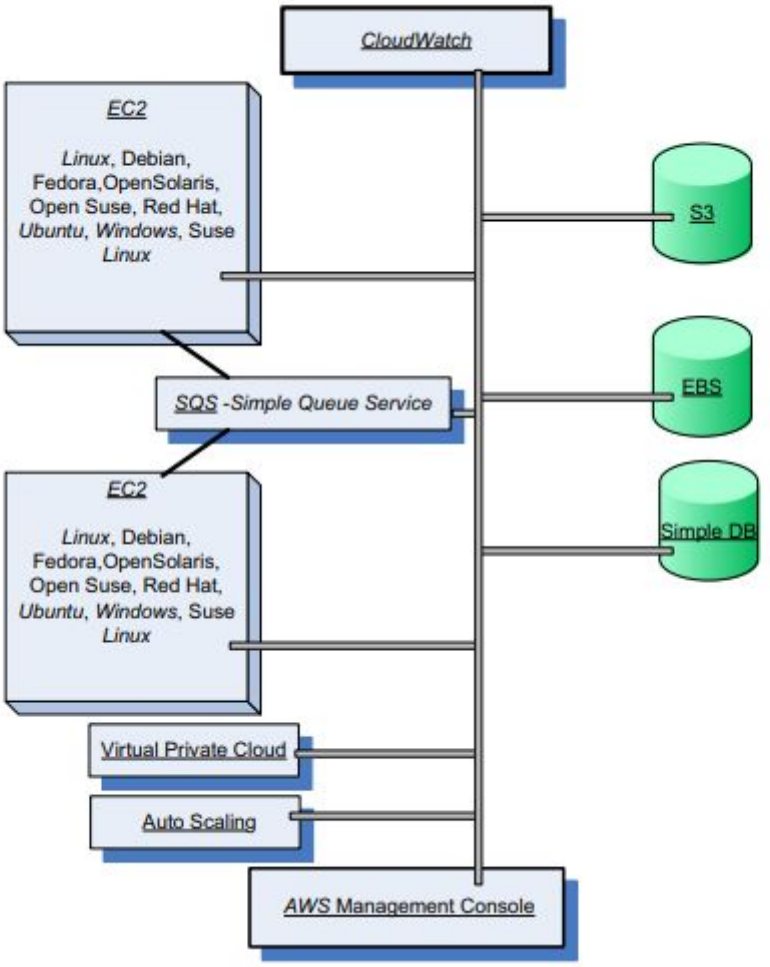
Network Centric Content

Content: any type or volume of media, be it static or dynamic, monolithic or modular, live or stored, produced by aggregation, or mixed.
The “Future Internet” will be content-centric.
The creation and consumption of audio and visual content is likely to transform the Internet to support increased quality in terms of resolution, frame rate, color depth, stereoscopic information.

Characteristics

Data-intensive: large scale simulations in science and engineering require large volumes of data. Multimedia streaming transfers large volume of data.
Network-intensive: transferring large volumes of data requires high bandwidth networks.
Low-latency networks for data streaming, parallel computing, computation steering.
The systems are accessed using thin clients running on systems with limited resources, e.g., wireless devices such as smart phones and tablets.
The infrastructure should support some form of workflow management.

b) **Describe briefly with an architectural diagram the services offered by AWS, accessible from the AWS Management Console.**



- Elastic Compute Cloud (EC2)1 is a Web service with a simple interface for launching instances of an application under several operating systems, such as several Linux distributions, Microsoft Windows Server 2003 and 2008, OpenSolaris, FreeBSD, and NetBSD.
- Simple Storage System (S3) is a storage service designed to store large objects. It supports a minimal set of functions: write, read, and delete.
- Elastic Block Store (EBS) provides persistent block-level storage volumes for use with Amazon EC2 instances. A volume appears to an application as a raw, unformatted, and reliable physical disk; the size of the storage volumes ranges from one gigabyte to one terabyte.

- Simple DB is a nonrelational data store that allows developers to store and query data items via Web services requests. It supports store-and-query functions traditionally provided only by relational databases.
- Simple Queue Service (SQS) is a hosted message queue. SQS is a system for supporting automated workflows; it allows multiple Amazon EC2 instances to coordinate their activities by sending and receiving SQS messages
- CloudWatch is a monitoring infrastructure used by application developers, users, and system administrators to collect and track metrics important for optimizing the performance of applications and for increasing the efficiency of resource utilization.
- Virtual Private Cloud (VPC) provides a bridge between the existing IT infrastructure of an organization and the AWS cloud. The existing infrastructure is connected via a virtual private network (VPN) to a set of isolated AWS compute resources
- Auto Scaling exploits cloud elasticity and provides automatic scaling of EC2 instances. The service supports grouping of instances, monitoring of the instances in a group, and defining triggers and pairs of CloudWatch alarms and policies, which allow the size of the group to be scaled up or down

2. a) Classify cloud computing delivery models based on the level of consumer control with neat diagrams.

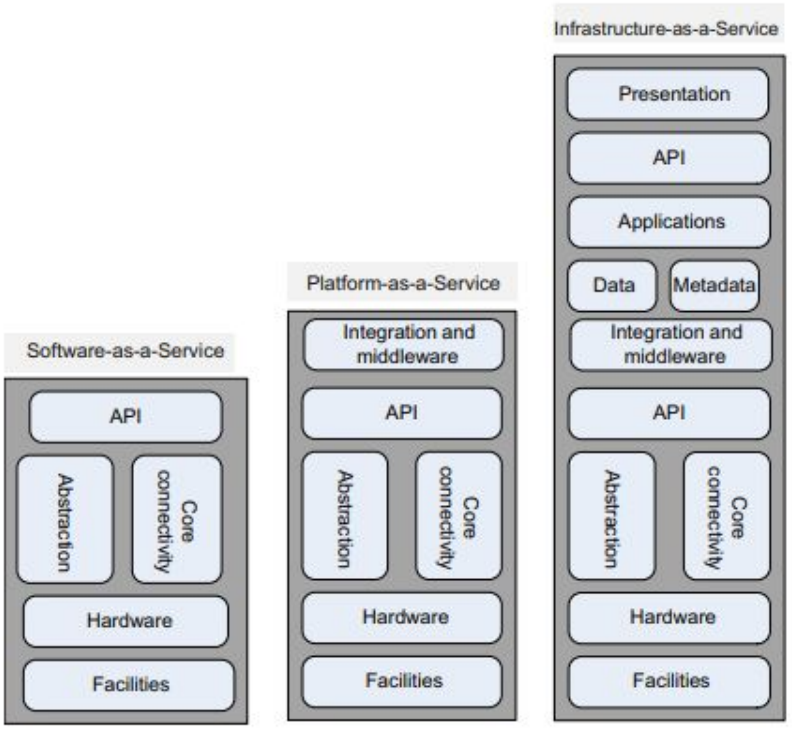


FIGURE 1.3

The structure of the three delivery models, *SaaS*, *PaaS*, and *IaaS*. *SaaS* gives users the capability to use applications supplied by the service provider but allows no control of the platform or the infrastructure. *PaaS* gives the capability to deploy consumer-created or acquired applications using programming languages and tools supported by the provider. *IaaS* allows the user to deploy and run arbitrary software, which can include operating systems and applications.

Infrastructure-as-a-Service (IaaS)

- The user is able to deploy and run arbitrary software, which can include operating systems and applications.
- The user does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of some networking components, e.g., host firewalls.
- Services offered by this delivery model include: server hosting, Web servers, storage, computing hardware, operating systems, virtual instances, load balancing, Internet access, and bandwidth provisioning.

Platform-as-a-Service (PaaS)

- Allows a cloud user to deploy consumer-created or acquired applications using programming languages and tools supported by the service provider.
- The user:
  - Has control over the deployed applications and, possibly, application hosting environment configurations.
  - Does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage.
- Not particularly useful when:
  - The application must be portable.
  - Proprietary programming languages are used.
  - The hardware and software must be customized to improve the performance of the application.

Software-as-a-Service (SaaS)

- Applications are supplied by the service provider.
- The user does not manage or control the underlying cloud infrastructure or individual application capabilities.
- Services offered include:
  - Enterprise services such as: workflow management, group-ware and collaborative, supply chain, communications, digital signature, customer relationship management (CRM), desktop software, financial management, geo-spatial, and search.
  - Web 2.0 applications such as: metadata management, social networking, blogs, wiki services, and portal services.
- Not suitable for real-time applications or for those where data is not allowed to be hosted externally.
- Examples: Gmail, Google search engine.

b) Describe with a diagram the architecture of Windows Azure.

3.3 *Microsoft Windows Azure* and online services

Azure and *Online Services* are, respectively, *PaaS* and *SaaS* cloud platforms from Microsoft. *Windows Azure* is an operating system, *SQL Azure* is a cloud-based version of the SQL Server, and *Azure AppFabric* (formerly .NET Services) is a collection of services for cloud applications.

Windows Azure has three core components (see Figure 3.3): *Compute*, which provides a computation environment; *Storage* for scalable storage; and *Fabric Controller*, which deploys, manages, and monitors applications; it interconnects nodes consisting of servers, high-speed connections, and switches.

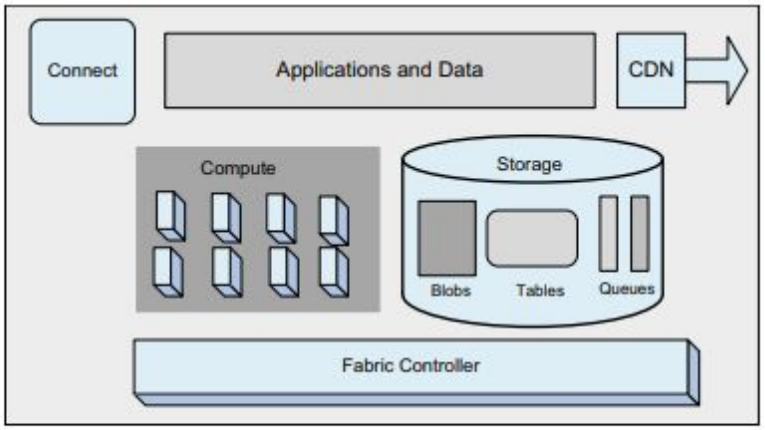


FIGURE 3.3

The components of *Windows Azure*: *Compute*, which runs cloud applications; *Storage*, which uses blobs, tables, and queues to store data; *Fabric Controller*, which deploys, manages, and monitors applications; CDN, which maintains cache copies of data; and *Connect*, which allows IP connections between the user systems and applications running on *Windows Azure*.

The Content Delivery Network (CDN) maintains cache copies of data to speed up computations

3. a) Describe using diagrams the basic workflow patterns.

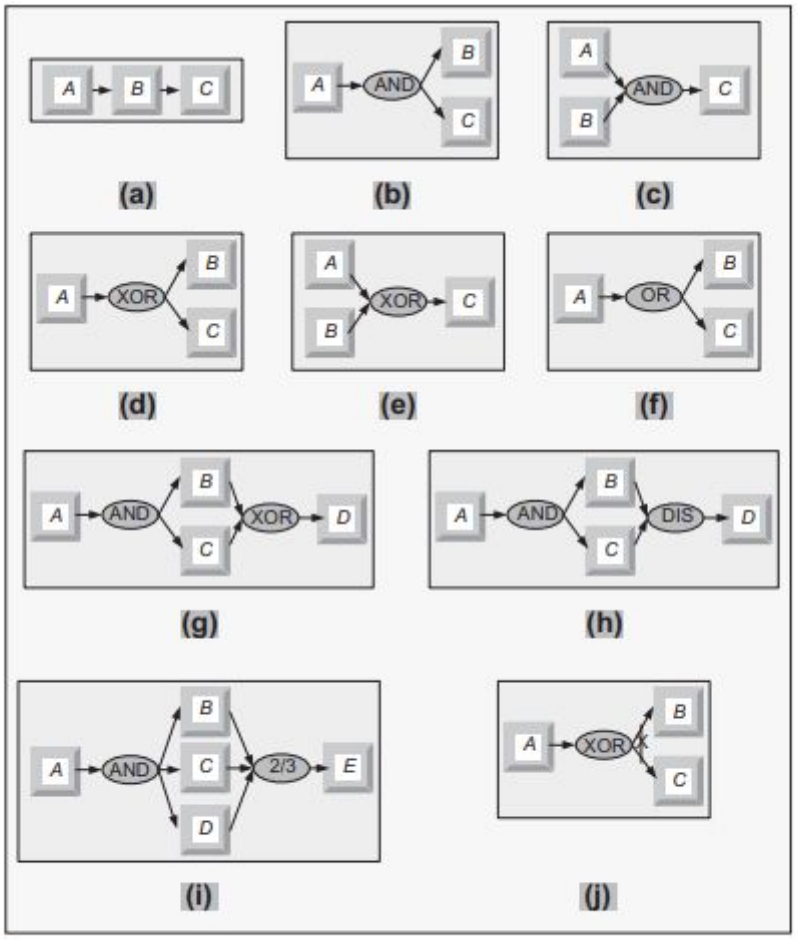


FIGURE 4.3

Basic workflow patterns. (a) *Sequence.* (b) *AND split.* (c) *Synchronization.* (d) *XOR split.* (e) *XOR merge.* (f) *OR split.* (g) *Multiple merge.* (h) *Discriminator.* (i) *N out of M join.* (j) *Deferred choice.*

Basic workflow patterns

- Workflow patterns - the temporal relationship among the tasks of a process
  - Sequence - several tasks have to be scheduled one after the completion of the other.
  - AND split - both tasks B and C are activated when task A terminates.
  - Synchronization - task C can only start after tasks A and B terminate.
  - XOR split - after completion of task A, either B or C can be activated.
  - XOR merge - task C is enabled when either A or B terminate.
  - OR split - after completion of task A one could activate either B, C, or both.
  - Multiple Merge - once task A terminates, B and C execute concurrently; when the first of them, say B, terminates, then D is activated; then, when C terminates, D is activated again.
  - Discriminator – wait for a number of incoming branches to complete before activating the subsequent activity; then wait for the remaining branches to finish without taking any action until all of them have terminated. Next, resets itself.

Basic workflow patterns (cont'd)

- N out of M join - barrier synchronization. Assuming that M tasks run concurrently, N (N<M) of them have to reach the barrier before the next task is enabled. In our example, any two out of the three tasks A, B, and C have to finish before E is enabled.
- Deferred Choice - similar to the XOR split but the choice is not made explicitly; the run-time environment decides what branch to take.

b) Illustrate with a diagram how the Cirrus platform helps in executing legacy Windows applications on the cloud.

In the Cirrus system a job has a description consisting of a prologue, a set of commands, and a set of parameters. The prologue sets up the running environment; the commands are sequences of shell scripts, including Azure-storage-related commands to transfer data between Azure blob storage and the instance. After the Windows Live ID service authenticates the user, it can submit and track a job through the portal provided by the Web role (see Figure 4.8). The job is added to a table called job registry. The execution of each job is controlled by a job manager instance that first scales the size of the worker based on the job configuration; then the parametric engine starts exploring the parameter space. If this is a test run, the parameter-sweeping result is sent to the sampling filter. Each task is associated with a record in the task table, and this state record is updated periodically by the worker instance running the task. The progress of the task is monitored by the manager. The

dispatch queue feeds into a set of worker instances. A worker periodically updates the task state in the task table and listens for any control signals from the manager.

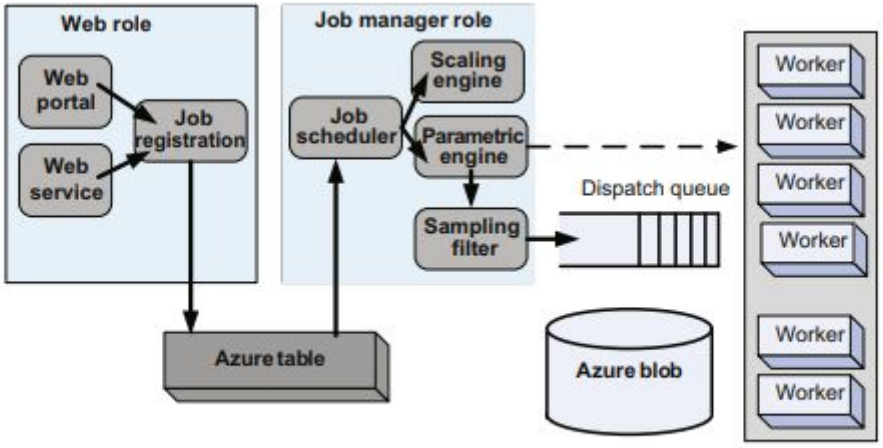


FIGURE 4.8

Cirrus, a general platform for executing legacy *Windows* applications on the cloud.

4. a) What is the concept of a task in workflow modeling? Identify the attributes of a task and describe different types of tasks.

Many cloud applications require the completion of multiple interdependent tasks; the description of a complex activity involving such an ensemble of tasks is known as a workflow

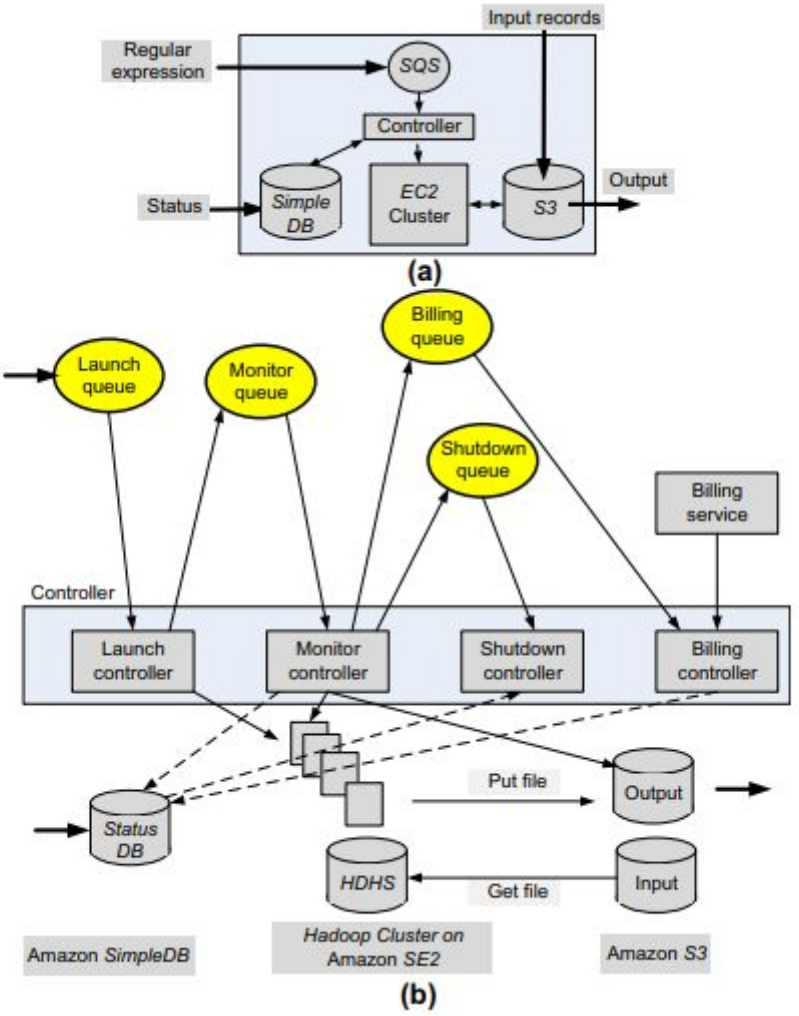
Workflow models are abstractions revealing the most important properties of the entities participating in a workflow management system. Task is the central concept in workflow modeling; a task is a unit of work to be performed on the cloud, and it is characterized by several attributes, such as:

• Name. A string of characters uniquely identifying the task.
• Description. A natural language description of the task. 4.4 Workflows: Coordination of Multiple Activities 105
• Actions. Modifications of the environment caused by the execution of the task.
• Preconditions. Boolean expressions that must be true before the action(s) of the task can take place.
• Post-conditions. Boolean expressions that must be true after the action(s) of the task take place.
• Attributes. Provide indications of the type and quantity of resources necessary for the execution of the task, the actors in charge of the tasks, the security requirements, whether the task is reversible, and other task characteristics.
• Exceptions. Provide information on how to handle abnormal events. The exceptions supported by a task consist of a list of pairs. The exceptions included in the task exception list are called anticipated exceptions, as opposed to unanticipated exceptions. Events not included in the exception list trigger replanning. Replanning means restructuring of a process or redefinition of the relationship among various tasks.

Types of Task:

1. A composite task is a structure describing a subset of tasks and the order of their execution.
2. A primitive task is one that cannot be decomposed into simpler tasks. A composite task inherits some properties from workflows; it consists of tasks and has one start symbol and possibly several end symbols. At the same time, a composite task inherits some properties from tasks; it has a name, preconditions, and post-conditions.
3. A routing task is a special-purpose task connecting two tasks in a workflow description. The task that has just completed execution is called the predecessor task; the one to be initiated next is called the successor task. A routing task could trigger a sequential, concurrent, or iterative execution. Several types of routing task exist: •
4. A fork routing task triggers execution of several successor tasks. Several semantics for this construct are possible: • All successor tasks are enabled. • Each successor task is associated with a condition. The conditions for all tasks are evaluated, and only the tasks with a true condition are enabled. • Each successor task is associated with a condition. The conditions for all tasks are evaluated, but the conditions are mutually exclusive and only one condition may be true. Thus, only one task is enabled. • Nondeterministic, k out of n > k successors are selected at random to be enabled.
5. A join routing task waits for completion of its predecessor tasks. There are several semantics for the join routing task: • The successor is enabled after all predecessors end. • The successor is enabled after k out of n > k predecessors end. • Iterative: The tasks between the fork and the join are executed repeatedly

b) Describe the steps in the workflow of the GrepTheWeb application with a diagram.



1. The startup phase. Creates several queues – launch, monitor, billing, and shutdown queues. Starts the corresponding controller threads. Each thread periodically polls its input queue and, when a message is available, retrieves the message, parses it, and takes the required actions.
2. The processing phase. This phase is triggered by a StartGrep user request; then a launch message is enqueued in the launch queue. The launch controller thread picks up the message and executes the launch task; then, it updates the status and time stamps in the Amazon Simple DB domain. Finally, it enqueues a message in the monitor queue and deletes the message from the launch queue. The processing phase consists of the following steps:

- a. The launch task starts Amazon EC2 instances. It uses a Java Runtime Environment preinstalled Amazon Machine Image (AMI), deploys required Hadoop libraries, and starts a Hadoop Job (run Map/Reduce tasks).
- b. Hadoop runs map tasks on Amazon EC2 slave nodes in parallel. A map task takes files from Amazon S3, runs a regular expression, and writes the match results locally, along with a description of up to five matches. Then the

combine/reduce task combines and sorts the results and consolidates the output.

- c. Final results are stored on Amazon S3 in the output bucket.

3. The monitoring phase. The monitor controller thread retrieves the message left at the beginning of the processing phase, validates the status/error in Amazon Simple DB, and executes the monitor task. It updates the status in the Amazon Simple DB domain and enqueues messages in the shutdown and billing queues. The monitor task checks for the Hadoop status periodically and updates the Simple DB items with status/error and the Amazon S3 output file. Finally, it deletes the message from the monitor queue when the processing is completed.

4. The shutdown phase. The shutdown controller thread retrieves the message from the shutdown queue and executes the shutdown task, which updates the status and time stamps in the Amazon Simple DB domain. Finally, it deletes the message from the shutdown queue after processing. The shutdown phase consists of the following steps:

- a. The shutdown task kills the Hadoop processes, terminates the EC2 instances after getting EC2 topology information from Amazon Simple DB, and disposes of the infrastructure.
- b. The billing task gets the EC2 topology information, Simple DB usage, and S3 file and query input, calculates the charges, and passes the information to the billing service.

5. The cleanup phase. Archives the Simple DB data with user info.

6. User interactions with the system. Get the status and output results. The GetStatus is applied to the service endpoint to get the status of the overall system (all controllers and Hadoop) and download the filtered results from Amazon S3 after completion.

1. a) Draw the NIST cloud computing reference model and discuss the function of the entities involved.

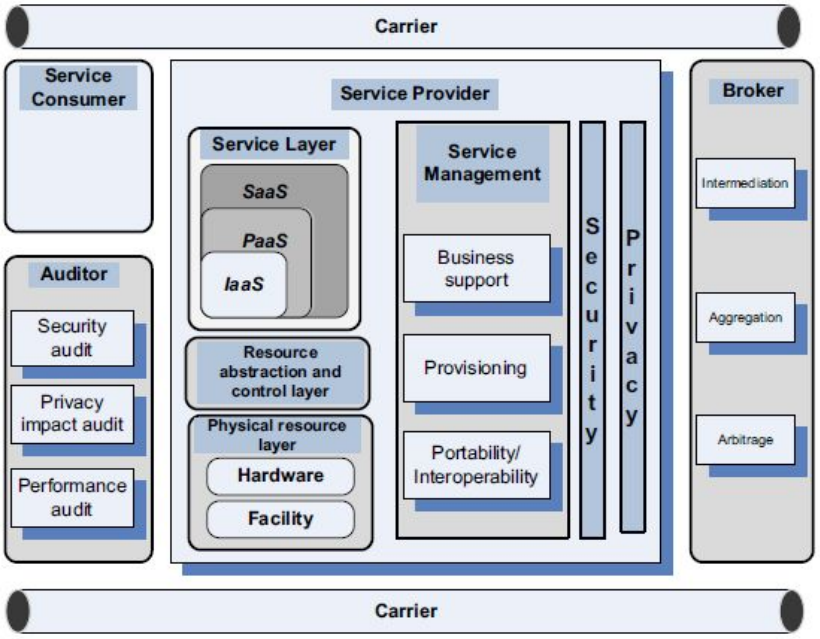


FIGURE 1.2

The entities involved in service-oriented computing and, in particular, in cloud computing, according to NIST. The carrier provides connectivity among service providers, service consumers, brokers, and auditors.

According to the NIST reference model in Figure 1.2 [260], the entities involved in cloud computing are the service consumer, the entity that maintains a business relationship with and uses service from service providers; the service provider, the entity responsible for making a service available to service consumers; the carrier, the intermediary that provides connectivity and transport of cloud services between providers and consumers; the broker, an entity that manages the use, performance, and delivery of cloud services and negotiates relationships between providers and consumers; and the auditor, a party that can conduct independent assessment of cloud services, information system operations, performance, and security of the cloud implementation. An audit is a systematic evaluation of a cloud system that measures how well it conforms to a set of established criteria. For example, a security audit evaluates cloud security, a privacy-impact audit evaluates cloud privacy assurance, and a performance audit evaluates cloud performance.

b) Describe and compare the three storage services offered by AWS.

S3 – Simple Storage System

- Service designed to store large objects; an application can handle an unlimited number of objects ranging in size from 1 byte to 5 TB.
- An object is stored in a bucket and retrieved via a unique, developer-assigned key; a bucket can be stored in a Region selected by the user.
- Supports a minimal set of functions: write, read, and delete; it does not support primitives to copy, to rename, or to move an object from one bucket to another.
- The object names are global.
- S3 maintains for each object: the name, modification time, an access control list, and up to 4 KB of user-defined metadata.

S3 (cont'd)

- Authentication mechanisms ensure that data is kept secure.
- Objects can be made public, and rights can be granted to other users.
- *S3* computes the MD5 of every object written and returns it in a field called ETag.
- A user is expected to compute the MD5 of an object stored or written and compare this with the ETag; if the two values do not match, then the object was corrupted during transmission or storage.

Elastic Block Store (EBS)

- Provides persistent block level storage volumes for use with *EC2* instances; suitable for database applications, file systems, and applications using raw data devices.
- A volume appears to an application as a raw, unformatted and reliable physical disk; the range 1 GB -1 TB.
- An *EC2* instance may mount multiple volumes, but a volume cannot be shared among multiple instances.
- EBS supports the creation of snapshots of the volumes attached to an instance and then uses them to restart the instance.
- The volumes are grouped together in Availability Zones and are automatically replicated in each zone.

SimpleDB

- Non-relational data store. Supports store and query functions traditionally provided only by relational databases.
- Supports high performance Web applications; users can store and query data items via Web services requests.
- Creates multiple geographically distributed copies of each data item.
- It manages automatically:
  - The infrastructure provisioning.
  - Hardware and software maintenance.
  - Replication and indexing of data items.
  - Performance tuning.

2 a) Illustrate with a diagram, responsibility sharing between the user and cloud service provider.

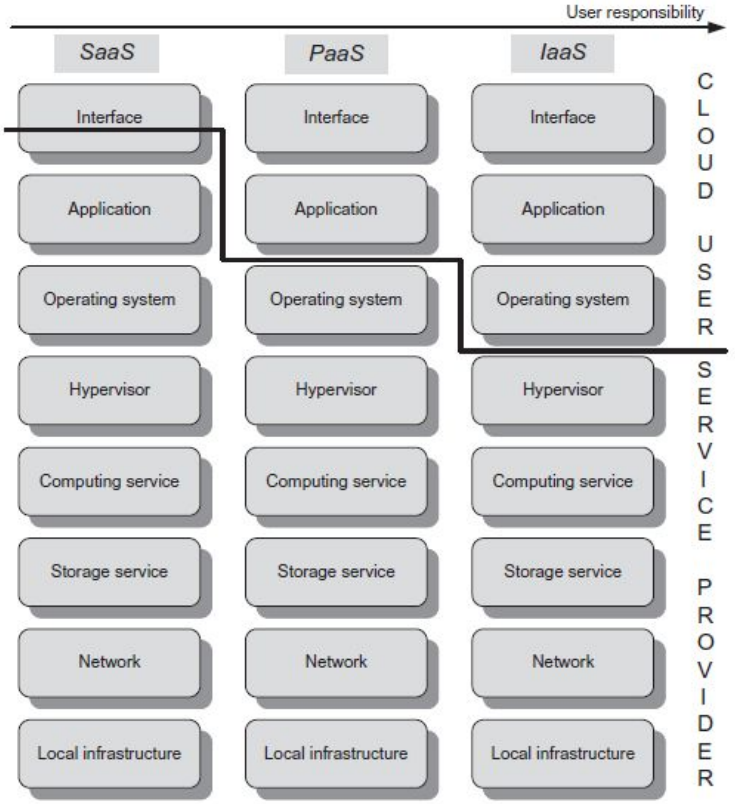


FIGURE 3.7

The limits of responsibility between a cloud user and the cloud service provider.

The level of user control over the system in IaaS is different from PaaS. IaaS provides total control,whereas PaaS typically provides no control. Consequently, IaaS incurs administration costs similar to a traditional computing infrastructure, whereas the administrative costs are virtually zero for PaaS.It is critical for a cloud user to carefully read the SLA and to understand the limitations of the liability a cloud provider is willing to accept. In many instances the liabilities do not apply to damages caused by a third party or to failures attributed either to the customer's hardware and software or to hardware and software from a third party.

The limits of responsibility between the cloud user and the cloud service provider are different for the three service-delivery models, as we can see in Figure 3.7. In the case of SaaS the user is partially responsible for the interface; the user responsibility increases in the case of PaaS and includes the interface and the application. In the case of IaaS the user is responsible for all the events occurring in the virtual machine running the application. For example, if a distributed denial-of-service attack (DDoS) causes the entire IaaS infrastructure to fail, the cloud service provider is responsible for the consequences of the attack. The user is responsible if the DDoS affects only several instances, including the ones running the user application. A recent posting describes the limits of responsibility illustrated in Figure 3.7 and argues that security should be a major concern for IaaS cloud users.

b) Illustrate with necessary diagrams the RAID-5 technology, and show with an example how it can be used to provide reliable data storage on the cloud.

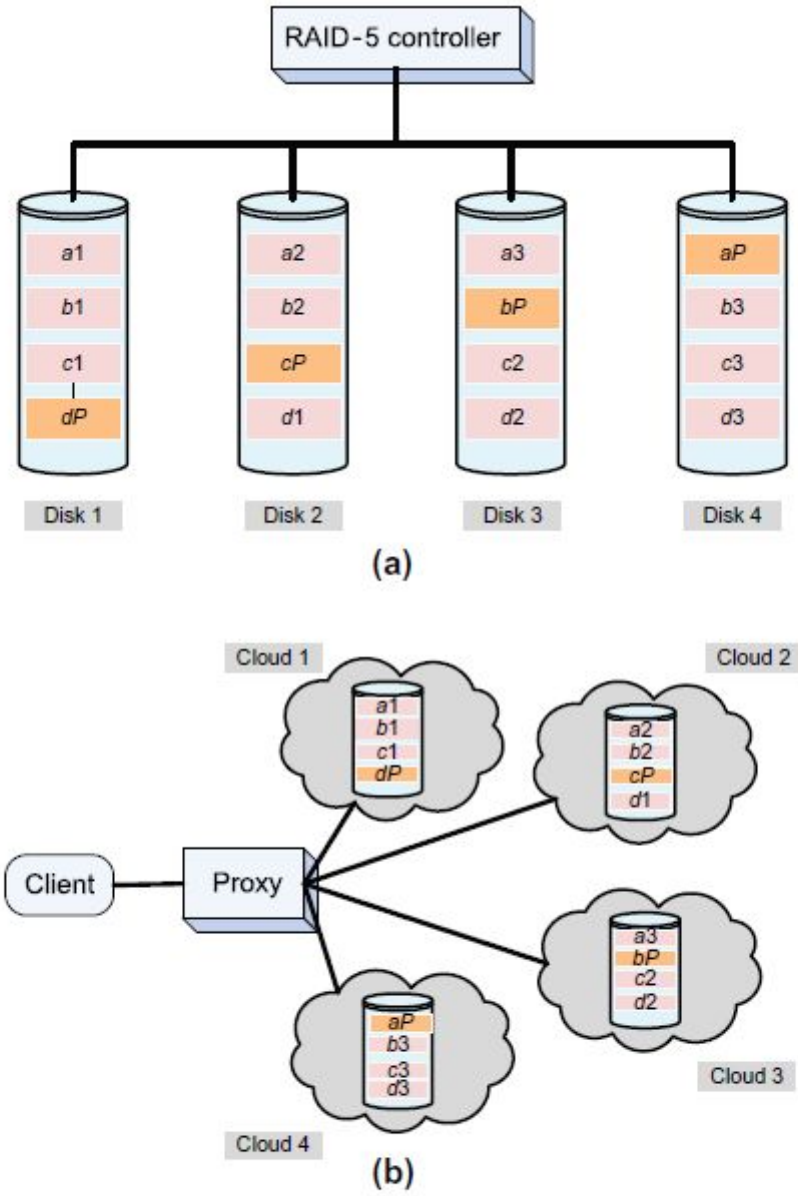


FIGURE 3.5

(a) A (3, 4) RAID-5 configuration in which individual blocks are stripped over three disks and a parity block is added; the parity block is constructed by XOR-ing the data blocks (e.g., $aP = a1 \text{XOR} a2 \text{XOR} a3$). The parity blocks are distributed among the four disks: aP is on disk 4, bP on disk 3, cP on disk 2, and dP on disk 1. (b) A system that strips data across four clouds; the proxy provides transparent access to data.

A RAID-5 system uses block-level stripping with distributed parity over a disk array, as shown in Figure 3.5(a); the disk controller distributes the sequential blocks of data to the physical disks and computes a parity block by bit-wise XOR-ing of the data blocks. The parity block is written on a different disk for each file to avoid the bottleneck possible when all parity blocks are written to a dedicated disk,as is done in the case of RAID-4 systems.

This technique allows us to recover the data after a single disk loss. For example, if Disk 2 in Figure 3.5 is lost, we still have all the blocks of the third file, c1,c2, and c3, and we can recover the missing blocks for the others as follows:

a2 = (a1) XOR (aP) XOR (a3)
b2 = (b1) XOR (bP) XOR (b3)
d1 = (dP) XOR (d2) XOR (d3)

Obviously, we can also detect and correct errors in a single block using the same procedure. The RAID controller also allows parallel access to data (for example, the blocks a1, a2, and a3 can be read and written concurrently) and it can aggregate multiple write operations to improve performance.The system in Figure 3.5(b) strips the data across four clusters. The access to data is controlled by a proxy that carries out some of the functions of a RAID controller, as well as authentication and other security-related functions. The proxy ensures before-and-after atomicity as well as all-or-nothing atomicity for data access; the proxy buffers the data, possibly converts the data manipulation commands,optimizes the data access (e.g., aggregates multiple write operations), converts data to formats specific to each cloud, and so on.

3. a) Justify with diagrams the parallel between workflows and programs.

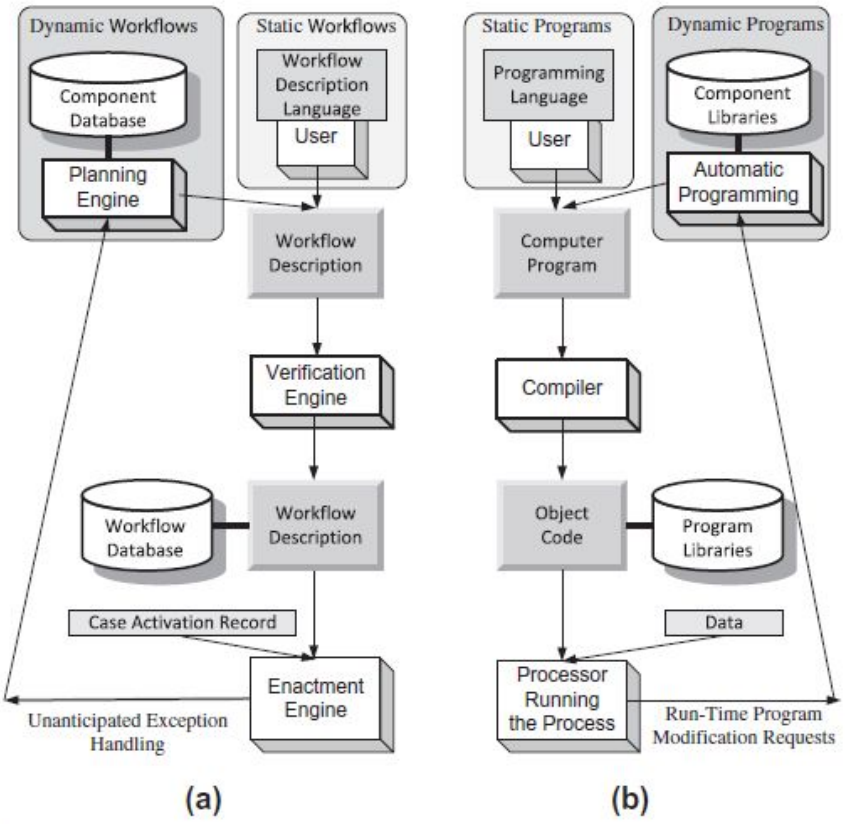


FIGURE 4.1

A parallel between workflows and programs. (a) The life cycle of a workflow. (b) The life cycle of a computer program. The workflow definition is analogous to writing a program. Planning is analogous to automatic program generation. Verification corresponds to syntactic verification of a program. Workflow enactment mirrors the execution of a program. A static workflow corresponds to a static program and a dynamic workflow to a dynamic program.

The phases in the life cycle of a workflow are creation, definition, verification, and enactment. There is a striking similarity between the life cycle of a workflow and that of a traditional computer program,namely, creation, compilation, and execution (see Figure 4.1). The workflow specification by means of a workflow description language is analogous to writing a program. Planning is equivalent to automatic program generation. Workflow verification corresponds to syntactic verification of a program, and workflow enactment mirrors the execution of a compiled program.

A case is an instance of a process description. The start and stop symbols in the workflow description enable the creation and the termination of a case, respectively. An enactment model describes the steps taken to process a case. When a computer executes all tasks required by a workflow the enactment can be performed by a program called an enactment engine.

b) Describe the map-reduce philosophy with a neat diagram and an Example.

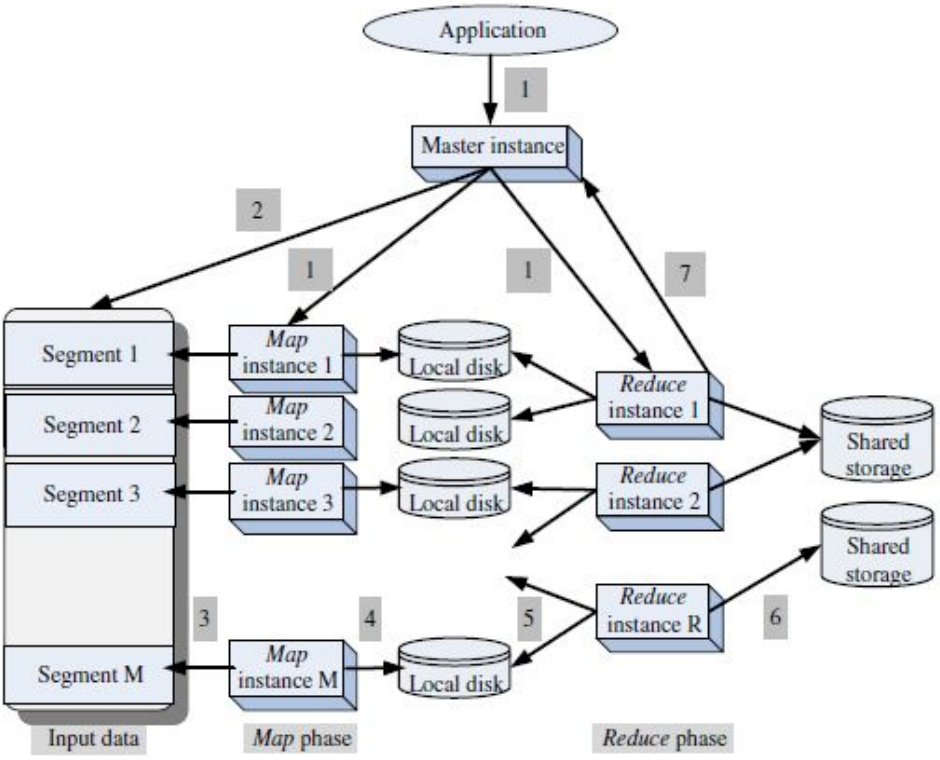


FIGURE 4.6

The *MapReduce* philosophy. (1) An application starts a *master instance* and *M* worker instances for the *Map* phase and, later, *R* worker instances for the *Reduce* phase. (2) The *master* partitions the input data in *M* segments. (3) Each *Map instance* reads its input data segment and processes the data. (4) The results of the processing are stored on the local disks of the servers where the *Map instances* run. (5) When all *Map instances* have finished processing their data, the *R Reduce instances* `read` the results of the first phase and merge the partial results. (6) The final results are written by the *Reduce instances* to a shared storage server. (7) The *master instance* monitors the *Reduce instances* and, when all of them report task completion, the application is terminated.

1. The run-time library splits the input files into M splits of 16 to 64 MB each, identifies a number N of systems to run, and starts multiple copies of the program, one of the system being a master and the others workers. The master assigns to each idle system either a Map or a Reduce task. The master makes O(M + R)scheduling decisions and keeps O(M × R) worker state vectors in memory. These considerations limit the size of M and R; at the same time, efficiency considerations require that M, R N.
2. A worker being assigned a Map task reads the corresponding input split, parses pairs, and passes each pair to a user-defined Map function. The intermediate pairs produced by the Map function are buffered in memory before being written to a local disk and partitioned into R regions by the partitioning function.
3. The locations of these buffered pairs on the local disk are passed back to themaster, who is responsible for forwarding these locations to the Reduce workers. A Reduce worker uses remote procedure calls to read the buffered data from the local disks of the Map workers; after reading all the intermediate data, it sorts it by the intermediate keys. For each unique intermediate key, the key and the corresponding set of intermediate values are passed to a user-defined Reduce function. The output of the Reduce function is appended to a final output file.
4. When all Map and Reduce tasks have been completed, the master wakes up the user program

4. a) Describe with necessary diagrams the organization of the Zookeeper service and the processing of read and write operations.

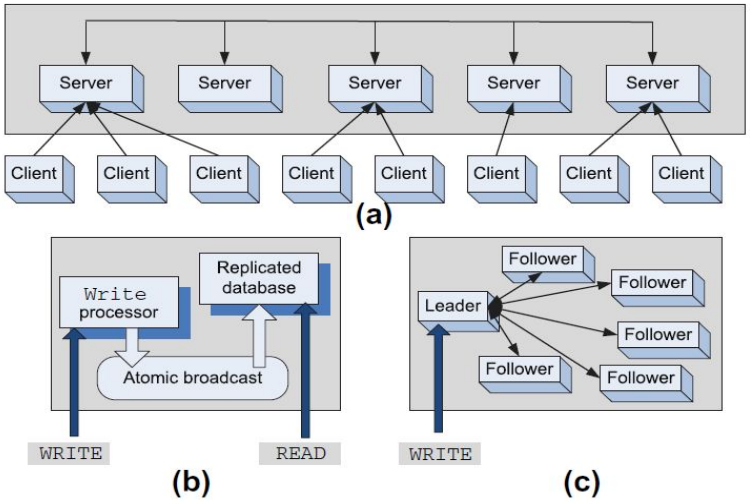


FIGURE 4.4

The *ZooKeeper* coordination service. (a) The service provides a single system image. Clients can connect to any server in the pack. (b) Functional model of the *ZooKeeper* service. The replicated database is accessed directly by `read` commands; `write` commands involve more intricate processing based on atomic broadcast. (c) Processing a `write` command: (1) A server receiving the command from a client forwards the command

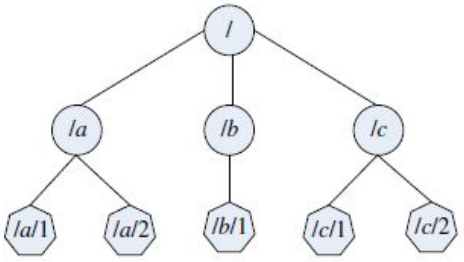


FIGURE 4.5

ZooKeeper is organized as a shared hierarchical namespace in which a name is a sequence of path elements separated by a backslash.

The system is organized as a shared hierarchical namespace similar to the organization of a file system. A name is a sequence of path elements separated by a backslash. Every name in Zookeper's namespace is identified by a unique path (see Figure 4.5).In ZooKeeper the znodes, the equivalent of the inodes of a file system, can have data associated with them. Indeed, the system is designed to store state information. The data in each node includes version numbers for the data, changes of ACLs,6 and time stamps. A client can set a watch on a znode and receive a notification when the znode changes. This organization allows coordinated updates. The data retrieved by a client also contains a version number. Each update is stamped with a number that reflects the order of the transition.

The data stored in each node is read and written atomically. A read returns all the data stored in a znode, whereas a write replaces all the data in the znode. Unlike in a file system, Zookeeper data,the image of the state, is stored in the server memory. Updates are logged to disk for recoverability, and writes are serialized to disk before they are applied to the in-memory database that contains the entire tree.

To reduce the response time, read requests are serviced from the local replica of the server that is connected to the client.When the leader receives a write request, it determines the state of the system where the write will be applied and then it transforms the state into a transaction that captures this new state.

b) Illustrate with a diagram how the BigJob software system can be used for the execution of loosely coupled workloads on the Azure platform.

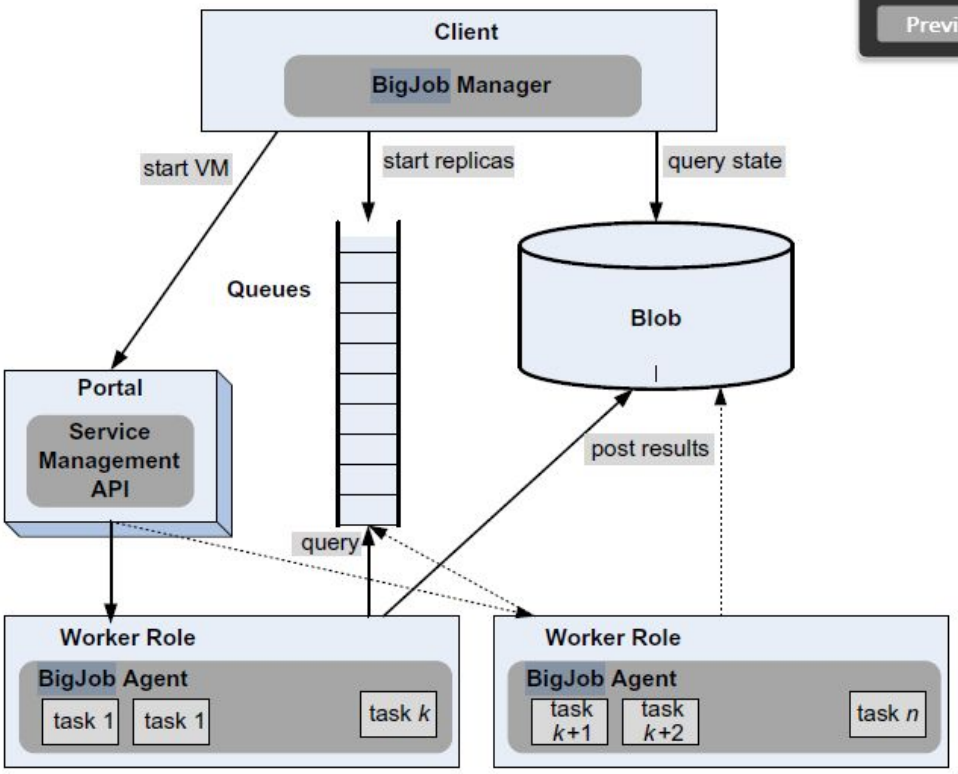


FIGURE 4.9

The execution of loosely coupled workloads using the Azure platform.

Figure 4.9 illustrates the use of a software system called BigJob to decouple resource allocation from resource binding for the execution of loosely coupled workloads on an Azure platform [224]. This software eliminates the need for the application to manage individual VMs. The results of measurements show a noticeable overhead for starting VMs and for launching the execution of an application task on a remote resource. Increasing the computing power of the VM decreases the completion time for long-running tasks.