8th International Conference
on Information Technology Engineering, Computer Sciences and Telecommunication of Iran
هشتمین همایش بین المللی مهندسی فناوری اطلاعات کامپیوتر و مخابرات ایران

Investigating cloud computing and analyzing its impact on information technology

Somayye Nasiri

Department of Computer Engineering Zanjan Branch,Islamic Azad University,Zanjan,Iran

ABSTRACT

Cloud computing is a model for easy and unlimited access to shared computing resources. These resources can be provided quickly, with minimal effort and minimal interaction with the service provider. Today, cloud computing has become one of the important topics of information technology. Cloud computing or cloud network includes a set of computers, servers, hardware and software resources, as well as huge information processing and storage resources, which provide the possibility of using web-based systems and all the online systems we need. he does. This paper introduces the main concepts of cloud computing that provide scalable processing and storage services to be used to extract knowledge from large data repositories. The first part of this article defines cloud computing and discusses the main service and deployment models adopted by cloud providers. This section also describes a number of cloud platforms that can be used to implement applications and frameworks for distributed data analysis. The next part of the article discusses how to apply cloud computing technologies in the implementation of distributed data analysis systems. This paper first defines the basic needs to be answered by a distributed data analytics system, and then examines how a cloud platform can be used to answer such needs.

Keywords: *OpenStack, Information technology, web service, cloud computing, data.*

1. Introduction

An effective solution for extracting useful knowledge from large data repositories in a reasonable period of time is to apply parallel and distributed data mining methods. It is also necessary and useful to work with data analysis environments that allow access, management and effective and efficient data mining of such repositories. For example, a scientist can use a data analysis environment to run complex data mining algorithms, validate models, and compare and share results with colleagues around the world. In recent years, clouds have emerged as effective computing platforms to face the challenges of retrieving information from large data repositories, as well as providing suitable and efficient data analysis environments for researchers and companies. From a user's point of view, the cloud is actually an abstract concept of infinitely scalable and remote computing and storage resources. From the point of view of implementing this view, cloud systems are based on a large set of computer resources that are located somewhere in the cloud and are assigned to applications based on demand. Therefore, cloud computing can be defined as a distributed paradigm in which all resources are dynamically scalable and often virtualized and offered as a service over the Internet. Virtualization is a software method that implements the separation of physical computing infrastructure and allows different virtual resources to be created on the same hardware. Virtualization is a basic method that enables cloud computing to run different operating environments and multiple applications simultaneously on a single server. Unlike other distributed computing models, users in cloud computing do not need to have knowledge and expertise or control the infrastructure that supports them in the cloud. A number of characteristics that define cloud applications, services, data and infrastructure:

• Remote hosting: Services and/or data are hosted on an infrastructure that is located remotely.

• Ubiquitous: Services or data are available from anywhere.

• Pay-per-use: Cloud computing is a metered utility that, like traditional utilities such as gas and electricity, only pays for the amount consumed.

We can also use the National Institute of Standards and Technology (NIST) general definition of cloud computing to highlight its main features (Mell and Grance, 2011): "Cloud computing is a model for convenient and on-demand access over a network to a pool of configurable and shared resources (eg, networks, servers, repositories, applications, and services) that can be rapidly provisioned and distributed with minimal management effort or provider involvement". Based on the definition provided by NIST, we can define five basic characteristics of cloud computing systems as on-demand, self-service, wide network access, pool of resources, fast flexibility and scalable service.

Cloud systems can be classified based on service models (software as a service, platform as a service, and infrastructure as a service) and deployment models (public cloud, private cloud, and hybrid cloud).

2. Service models in cloud computing

Cloud computing vendors provide their services according to three main models: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS).

Software as a service defines a delivery model where software and data are provided to the customer as ready-to-use services over the Internet. The software and associated data are hosted by the providers and can be accessed by customers without the need for additional hardware or software. Additionally, customers typically pay a monthly/annual fee without the need to purchase additional infrastructure or software ownership. Webmail systems (such as Gmail), calendars (Yahoo Calendar), document management (Microsoft Office 365), photo manipulation (Photoshop Express), customer relationship management (Salesforce), etc. are common examples of SaaS applications.

In the platform-as-a-service model, cloud vendors provide customers with a computing platform that typically includes databases, application servers, and a development environment for creating, testing, and running applications. Developers can focus only on deploying applications because cloud providers are responsible for maintaining and optimizing the environment and infrastructure. Therefore, in line with the development of applications, customers use a set of environmental services that are modular and can be easily integrated with each other. Typically, applications are developed as SaaS and ready-to-use. Google App Engine, Microsoft Azure and Salesforce.com are examples of Paas cloud environments. Finally, Infrastructure as a Service is an outsourcing model where customers rent resources such as CPUs, disks, or more complex resources such as virtualized servers or operating systems to support their operations (eg Amazon EC2 and RackSpaceCloud). Typically, users of the IaaS model have systems and network management skills, as they must deal with configuration, operation, and maintenance tasks. Compared to the PaaS approach, the IaaS model has high system management costs for users; On the other hand, IaaS enables the customization of the execution environment completely. Developers can scale their services by adding or subtracting virtual machines.

In Table 1, three service models are compared in terms of flexibility, scalability, portability, security, maintenance and cost.

Table 1: How SaaS, PaaS, and IaaS meet the needs of developers and end users.

IaaS	*PaaS*	*SaaS*	*Needs*
Developers have to create the servers that are supposed to host their applications and take responsibility for configuring the operating system and software modules that are supposed to run on these servers.	*Developers write, customize, and test their applications using platform-compatible libraries and support tools. Users can choose the type of virtual storage and computing resources that will be used to run their applications.*	*Users can customize the application interface and control its behavior, but they cannot decide what hardware and software components are used to support the execution of these applications.*	*Flexibility*
Developers can use new computing	*Similar to the SaaS model,*	*Infrastructure computing and*	*Scalability*

and storage resources, but their applications must be scalable and allow dynamic use of new resources.	*computing resources and infrastructure storage are usually scaled automatically.*	*storage resources are typically automatically scaled to match application demand, so users do not have to manually allocate resources. The result depends only on the degree of resilience that the cloud system provides.*	
If a provider allows the download of a virtual machine in a standard format, then it is possible to transfer to another provider in this model.	*Applications can be migrated to another provider only if the new provider shares the required platform services and tools with the previous provider.*	*Here, moving applications to other cloud providers can cause problems because some software and tools may not work on different systems. For example, application data may be in a format that the other provider cannot read.*	*Portability*
Developers must take care of security issues themselves in all areas, from the operating system to the application layers.	*The developer is responsible for the security of the codes and libraries used to create applications.*	*Users can control only a few of the security settings of their applications (for example, using https instead of http when accessing certain web pages). Additional security layers (eg, data replication) are hidden from users and managed directly by the system.*	*security*
Developers are responsible for maintaining all software	*Developers are only responsible for maintaining their own applications;*	*Users do not have to undertake maintenance tasks.*	*maintenance*

components, including the operating system; The hardware is maintained by the provider.	*Other software and hardware components are maintained by the provider.*		
Developers pay for all software and hardware resources they use.	*Developers pay for compute and storage resources, as well as licenses for libraries and tools used by their applications.*	*Users usually pay a monthly/yearly fee for using the software and do not pay any additional fees for the infrastructure.*	*cost*

3. Deployment models in cloud computing

Cloud computing services are offered according to three main deployment models: public, private and hybrid. A public cloud provider offers its services to the public over the Internet. Users of a public cloud have little or no control over the infrastructure technology. In this model, services can be provided for free or according to the pay-per-use policy. The main public providers such as Google, Microsoft and Amazon have dedicated data centers and manage and provide their services on these centers. A private cloud provider offers operations and capabilities as a service that are hosted on a company's intranet or in a remote data center. Because of the advanced security solutions and data control that the private cloud model provides and these solutions are not available in the public cloud model, most small and medium IT companies prefer the private cloud model. Finally, a hybrid cloud is actually a combination of two or more clouds (public or private) that remain separate but connected to each other. Companies can expand their private clouds using partner companies' private clouds or public clouds. In particular, by expanding private infrastructure with public cloud resources, it is possible to serve the most requests, better serve users' requests and implement strategies with maximum availability.

Figure 1 depicts the general architecture of a public cloud and its main components. Cloud computing services are provided using user equipment such as desktop computers, laptops, and smartphones. Users can access and interact with cloud-based services through these devices using browsers or desktop/companion applications. Business software and user data are run and stored on servers that are hosted in cloud data centers and provide computing resources and storage. The resources consist of thousands of servers and storage devices connected to each other through a network within the cloud. Data exchange between users and data centers using a wide network.

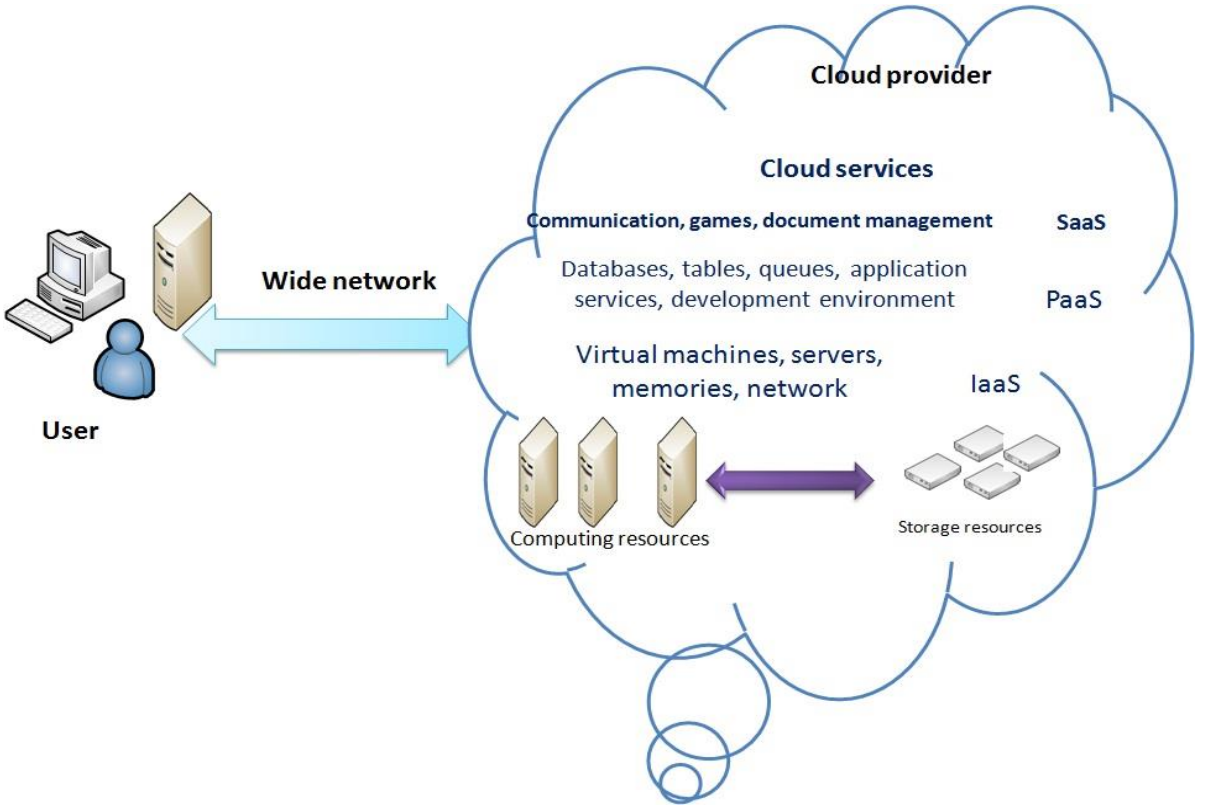

Figure 1: General architecture of a public cloud

Several technologies and standards can be used by different components in the architecture. For example, users can interact with cloud services through SOAP-based web services or other web services. HTML5 and Ajax technologies allow web interfaces to have the same visibility and interaction with cloud services as other desktop applications. The Open Cloud Computing Interface (OCCI) specifies how cloud providers can expose their computing, data, and network resources through standard interfaces. Another example is the Open Virtualization Format (OVF), which is used to package and distribute virtual appliances or software (for example, virtual operating systems) to run in virtual machines.

4. Cloud environments

In this section, four examples are introduced as representatives of cloud environments, which are Microsoft Azure as an example of public PaaS, Amazon Web Service as the most famous public IaaS, OpenNebula and OpenStack as examples of private IaaS. These environments can be used to implement applications and frameworks for data analysis in the cloud.

- Microsoft Azure

Azure is an environment and set of cloud services that can be used to develop cloud-based applications or to enhance existing applications with cloud-based capabilities. The on-demand computing and storage resources that this platform provides are by leveraging the computing and storage power of Microsoft data centers. Azure is designed to support high-availability and dynamically scalable services that fit a pay-as-you-go model.

The Azure platform can be used to store large databases, run large volumes of operational computing, and develop SaaS applications that target end users.

Microsoft Azure consists of three services/parts as shown in Figure 2.

• Computing part: computing environment that is used to run cloud applications. Each application is structured around roles: the web role, for web-based applications; Worker role, for executive applications; Virtual machine role, for virtual machine images.

• Storage: Provides scalable storage resources for managing text and binary data (Blobs), non-relational tables (Tables), queues for asynchronous communication between components (Queues) and virtual disks (Disks).

• Component controller part: The purpose of this part is to create a connected network of nodes in the physical machines of a data center. Compute and storage services are built on top of this component.

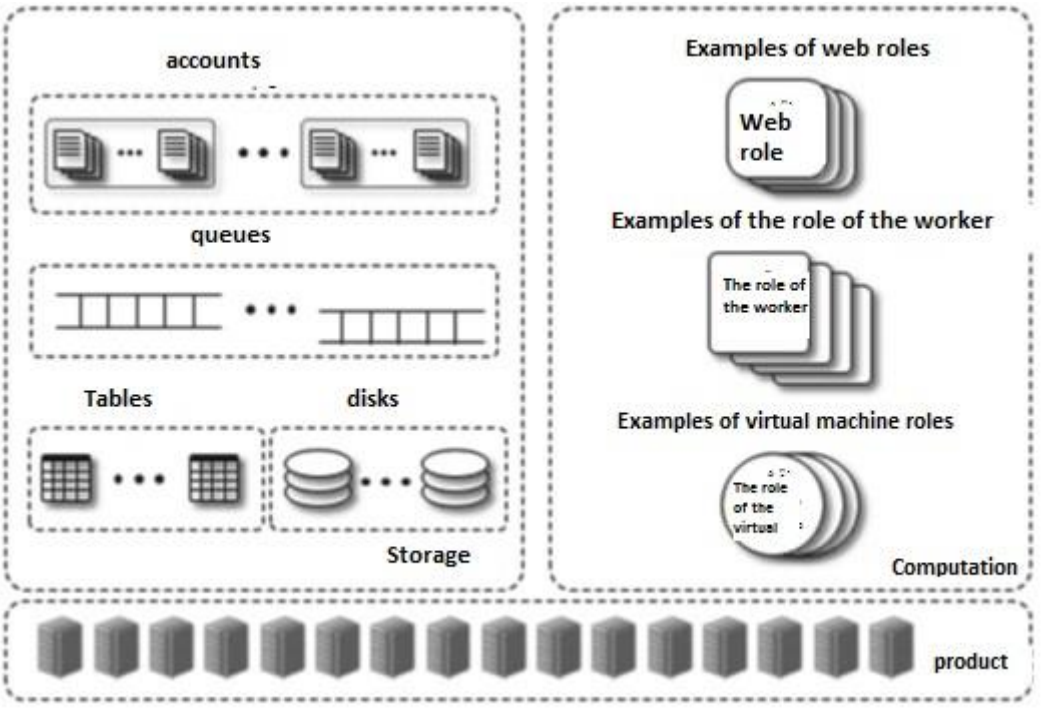

Figure 2: Microsoft Azure

Microsoft Azure provides standard interfaces that allow developers to interact with its services. In addition, developers can use integrated development environments (IDEs) such as Microsoft Visual Studio and Eclipse to more easily design and publish Azure applications.

- Amazon Web Services

Amazon provides computing and storage resources of its IT infrastructure to developers in the form of web services. Amazon Web Services (AWS) is actually a large set of cloud services that can be configured by users to create SaaS applications or integrate conventional software with cloud capabilities (Figure 3). Because Amazon provides SDKs for

programming purposes with different languages and platforms (eg Java, .Net, PHP, and Android), interacting with Amazon services is simple.

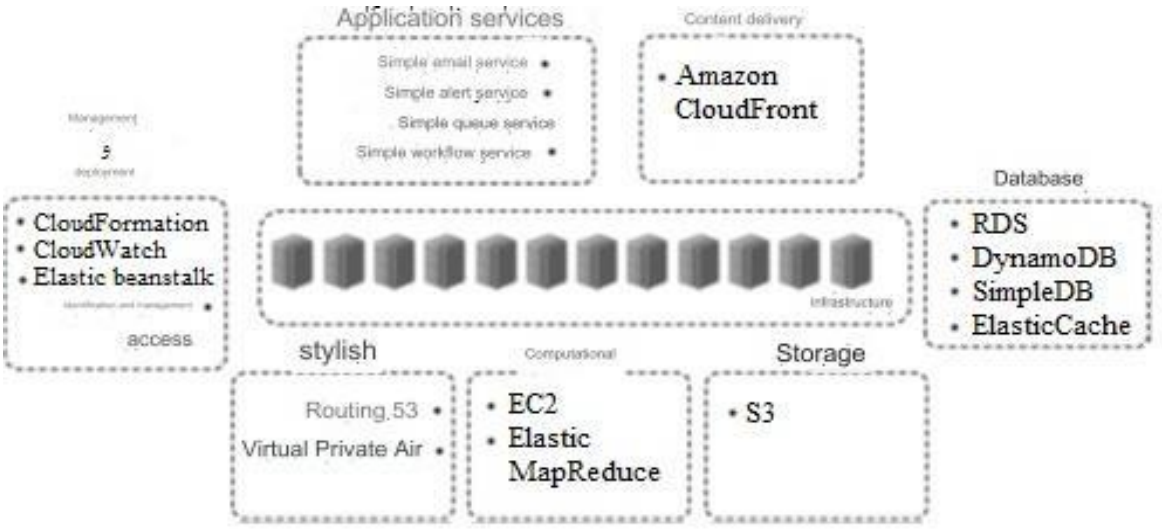

Figure 3: Amazon Web Services

AWS includes the following core services:

• Computational: elastic cloud computing (EC2) allows the creation and execution of virtual servers; Amazon Elastic MapReduce is for creating and running MapReduce applications.

• Storage: A simple storage service (S3) that allows storing and retrieving data over the Internet.

• Database: Relational Database Service (RDS) for relational tables; DynamoDB for non-relational tables; SimpleDB for managing small databases; Elastic cache for caching data.

• Network: Route 53 which is a DNS Web Service; A virtual private cloud that is used to implement a virtual network.

• Deployment and management: CloudFormation to create a set of ready-to-use virtual machines with pre-installed software (eg web applications); CloudWatch to monitor AWS resources; Elastic Beanstalk for deploying and running client applications written in Java, PHP, and other languages; Identity and access management for security control of access to AWS resources and services.

• Content delivery: Amazon CloudFront makes it easy to distribute content over the public network.

• Application Services: A simple email service that provides a basic email sending service. A simple notification service to notify users; A simple queue service that implements a queue of messages; A simple workflow service for implementing workflow-based applications.

Although Amazon is known as the first IaaS provider (based on its EC2 and S3-based services), it also operates as a PaaS provider today with services such as Elastic Beanstalk.

- OpenNebula

OpenNebula (Sotomayor et al., 2009) is actually the main open source framework used to create private and hybrid clouds. The main component of the OpenNebula architecture is the kernel, which creates and controls virtual machines by connecting them to a virtual network environment (Figure 4). In addition, the kernel interacts with pluggable components called drivers to perform storage, networking, and virtualization operations. In this way, OpenNebula is independent of the underlying infrastructure and offers a uniform management environment.

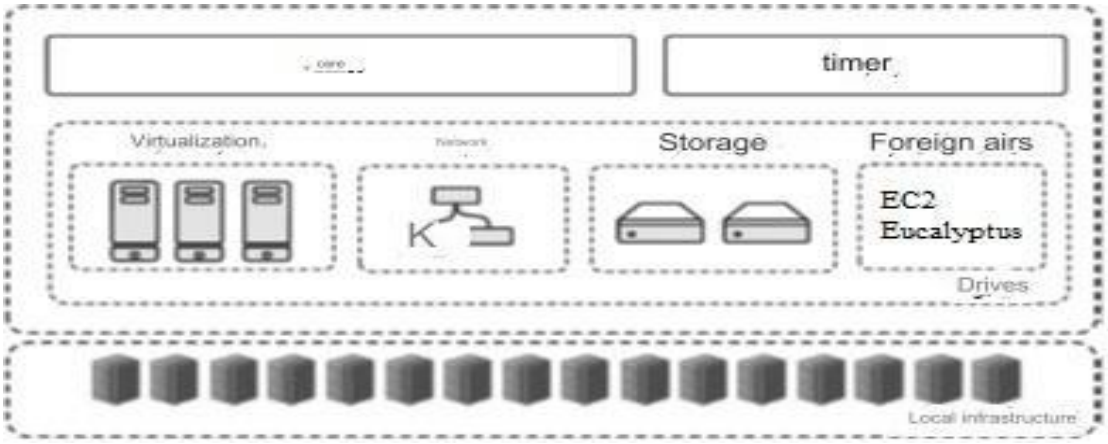

Figure 4: OpenNebula architecture

The kernel also supports the deployment of services, which are a collection of interconnected components (eg, web server, database) running on multiple virtual machines. Another component is the scheduler, which is responsible for allocating virtual machines on physical servers. For this purpose, the scheduler interacts with the kernel through appropriate built-in commands. OpenNebula can be implemented as a hybrid cloud that interacts with external clouds using special drivers. In this way, local infrastructure can be integrated with public cloud computing and storage resources. Currently, OpenNebula includes drivers for using Amazon EC2 resources and another open source framework called Eucalyptus.

- OpenStack

openStack is actually a cloud operating system that provides management of a large set of processing, storage and networking resources in a data center through web-based interfaces. The intended system is designed, developed and distributed for four main purposes:

• Open source: OpenStack is released under the Apache rules.

• Open Design: Every six months, a design meeting is held to gather requirements and define new technical specifications for future releases.

• Open Development: A repository of source code is publicly available and maintained for development.

• Open community: Most decisions are made by the OpenStack community using a lazy conference model.

OpenStack's modular architecture is composed of four main components, as shown in Figure 5.

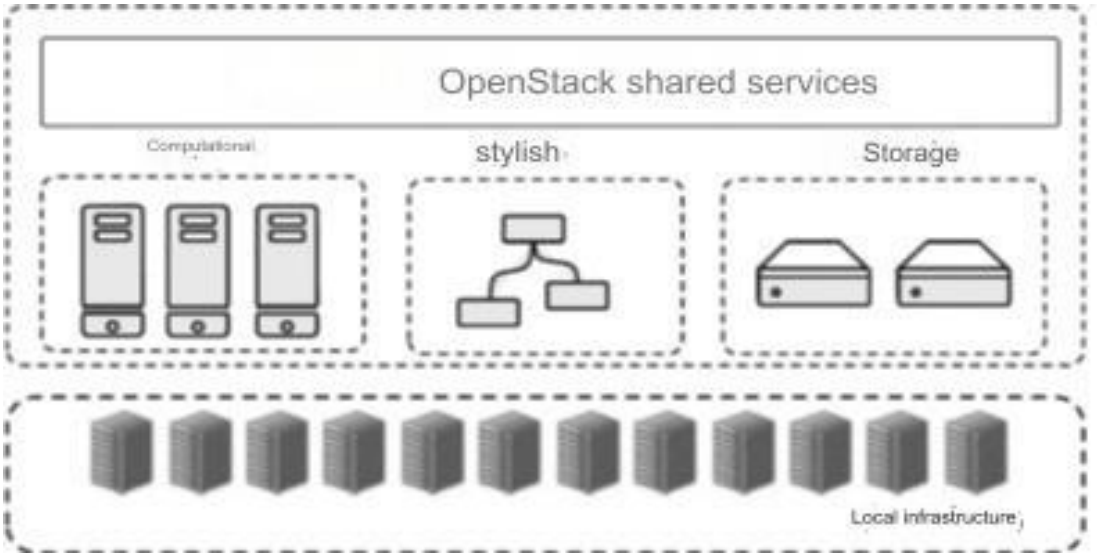

Figure 5: OpenStack

Compute provides on-demand virtual servers by managing available processing resources in the data center. Different technologies can be supported by this system (eg VMware, KVM) and can be horizontally scaled. The storage part of OpenStack provides a scalable and redundant storage system. It supports object and block storage, which allows storage and retrieval of objects and files in the data center, and block storage allows the creation, attachment, and detachment of server block devices. The networking part of OpenStack manages networks and IP addresses. Finally, OpenStack shared services are additional services provided for ease of use in the data center. For example, the identity service is for mapping users and servers, the image service is for managing image servers, and the database service is for providing a relational database.

5. Cloud computing systems for data-driven applications

Cloud systems can be effectively used to support data-centric applications because they provide flexible storage and processing services as well as software platforms for developing and running data analysis environments on top of such services. This section discusses cloud technologies that can be used to implement data analysis systems for KDD data centric applications. First, the definition of functional and non-functional requirements is determined that any KDD application that is a distributed data mining system must meet these requirements. Functional requirements specify which features of the system should be provided, and non-functional requirements refer to quality criteria that are more related to system performance.

- functional needs

The application requirements to be met by a distributed data analysis system are divided into two main categories: resource management requirements and application management requirements. Resource management requirements refer to requirements related to the management of all resources (data, tools, results) that may include a KDD application; Application management needs relate to the design and implementation of the applications themselves.

- Resource management

Resources that are of interest to KDD applications include data sources, knowledge discovery tools, and knowledge discovery results. Therefore, a distributed data analysis system should address the following resource management needs:

• Data management: Data sources can be in different formats such as relational databases, plain files, or semi-structured documents (eg XML files). The system must provide methods for storing and accessing such data sources independently of their specific format. In addition, metadata must be formally and formulaically defined, and descriptions of relevant information related to the data sources must be used (as For example, placement, format, accessibility, available display) to enable their effective access and use.

• Tool management: knowledge discovery tools that include algorithms and services for data selection, preprocessing, transformation, data mining and evaluation of results. The system should provide methods for accessing and using such tools independent of their specific implementation. Cloud data should be used to describe the most important features of KDD tools (such as their functionality, location, usage).

• Result management: The knowledge obtained from the results of the knowledge discovery stage is represented by a knowledge model (or data mining model). The system should provide methods for storing and accessing such models independent of their structure and format. Like data and tools, data mining models also need their content to be explained and interpreted by cloud data to enable their effective retrieval.

- Application management

A distributed data analysis system should provide effective methods for designing KDD centralized data applications (design management) and controlling their implementation (execution management).

•Design Management: The entire range of distributed data analysis applications, from simple data mining tasks to complex data mining patterns, are expressed as workflows. From a design perspective, three main categories of knowledge discovery applications can be defined: single-task applications, where discovery such as classification, clustering, or association rules is performed on a single data source; Wide parameter applications, where a dataset is analyzed using multiple instances of the same data mining algorithm but with different parameters. Workflow-based applications, where potentially complex knowledge discovery applications are characterized by graphs connecting data sources, data mining algorithms, and visualization tools. An overall system should provide environments for the effective design of all the aforementioned categories of data analysis applications.

• Execution management: The system should provide a distributed execution environment that supports the efficient execution of data analysis applications designed by users. Since applications range from single-task to complex knowledge discovery workflows, the execution environment must be able to handle such a variety of applications. In particular, the execution environment should provide the following capabilities, which are related to the various stages of the execution of an application program: access to data sources that have been data mined, allocation of computing resources required by these programs, execution of programs based on user-defined specifications. has done, which may be expressed as a

workflow; Display the results to the user. Also, the system should allow users to monitor the execution of applications.

- Non-functional needs

Non-functional requirements can be defined at three levels: user, architecture, and infrastructure. User requirements specify how the user should interact with the system; Architectural requirements specify which rules and regulations are used to inspire the design of the system architecture; Finally, infrastructure requirements describe the non-functional characteristics of the computing infrastructure.

- User needs

From a user's point of view, the following non-functional needs must be met:

• Usability: The system should be easily used by end users without needing to pass special training courses.

• Access from anywhere: Users must have the ability to access the system from anywhere using standard network technologies (for example, websites) either through desktop computers or through mobile devices.

• Data Protection: Data has a key and valuable role for users, so the system must protect the data that is data mined and have the knowledge to detect unauthorized access and intentional and accidental destruction.

- Architectural needs

The main non-functional requirements at the architectural level are divided into the following three categories:

• Service Orientation: The architecture should be designed as a set of networked software components (services) to implement various operational capabilities of the system to effectively enable their reuse, combination and interoperability.

• Openness and Extensibility: The architecture should be open to integration with new knowledge discovery tools and services. In addition, according to the principle of open and closed services, existing services should be open for development but closed for modification.

• Being independent from the infrastructure: the architecture should be designed as independent from the infrastructure as possible; In other words, system services must be able to exploit basic capabilities provided by different infrastructures.

- Infrastructure needs

Finally, from an infrastructure perspective, the following non-functional requirements must be met:

• Standardized availability: The infrastructure must open its services to standard technologies (such as web services) to make them usable as building blocks for creating high-level services or applications.

• Support for distributed and heterogeneous data: The infrastructure must be able to support very large databases with large dimensions stored in different formats in a data center or geographically distributed over many sites.

• Availability: The infrastructure must be in a functional state even if failures occur that affect a set of hardware/software resources. Thus, effective methods (such as redundancy) must be implemented to ensure reliable access to sensitive resources such as user data and applications.

• Scalability: The infrastructure must have the ability to support a growing workload (due to large data to process or heavy algorithms to execute) by dynamically allocating required resources (processors, storage and network resources) effectively and efficiently. Additionally, once the load is reduced, the infrastructure must release resources that are not needed.

• Efficiency: the infrastructure should minimize the consumption of resources for the execution of a task. In the case of parallel/distributed tasks, efficient allocation of processing nodes must be ensured. In addition, the infrastructure must be as efficient as possible to provide effective services.

• Security: The infrastructure must provide effective security methods to ensure data protection, identity management and privacy.

6. Cloud models for distributed data analysis

As discussed in the previous sections, cloud providers classify their services into three main categories: Software as a Service (SaaS), where any software or application running over the Internet is provided to customers as ready-to-use services; has been Platform as a Service (PaaS), providing platform services such as databases, application servers, or environments for building, testing, and running customer applications; Infrastructure as a Service (IaaS), providing resources such as CPUs, memory, and storage, to run virtualized systems on the cloud. Data analysis services for data-sensitive KDD applications may be implemented in one of the following ways:

• KDD as SaaS: In this case, a clear and well-defined data mining algorithm or a ready-to-use knowledge discovery tool is provided as a web service to end users who may use the service directly through a web browser do .

• KDD as PaaS: In this case, a support platform is provided for developers who intend to create their own applications or extend existing applications. Developers focus only on defining their own KDD applications without worrying about underlying infrastructure or distributed computing issues.

• KDD as IaaS: In this case, a set of virtual resources is provided as a computing infrastructure for developers to run their data mining applications or implement their KDD systems from scratch.

In all three scenarios above, the cloud plays the role of infrastructure provider, even in SaaS and PaaS cases the infrastructure layers can be transparent to end users.

As an example for the PaaS approach, Table 2 provides a summary of how the Microsoft Azure components and methods introduced in the previous sections can be effectively used to meet the proposed application requirements of a distributed data analysis system.

Table 2: Using Microsoft Azure to meet the application requirements of a distributed data analysis system

Microsoft Azure components	*functional needs*	
Different data formats: large binary objects (Blobs); non-relational tables (Tables), queues for data communication (Queues); Relational databases (SQL Database). *Metadata support: relational tables/databases for storing data descriptions; Client-definable fields can be added to data sources that contain blobs.*	*data*	*Resource management*
Implementation-independent access: tools that can appear as web services. Metadata support: relational tables/databases for storing data descriptions; Client-definable fields can be added to data sources that contain blobs; WSDL specifications for web services.	*tools*	
Model storage: Blobs to store results both textually and graphically. Metadata support: relational tables/databases for storing data descriptions; Client-definable fields can be added to data sources that contain blobs.	*Results*	
Single-task applications: Schedule the execution of a web service or binary tool on a worker role instance. Pervasive parameter applications: Schedule the concurrent execution of a set of web services or binary tools on a set of worker role instances. Workflow-based applications: Schedule the coordinated execution of a set of web services or binary tools on a set of worker role instances.	*design*	*Application management*
Storage resource access: managed by the storage layer. Allocation of computing resources: managed by the computing layer. Application monitoring and execution: Worker role instances/web services to execute individual tasks; Tables to store task information; Example web role to display monitoring information. Display results: tables/blobs to store/interpret inferential models; Example web role to display results.	*execution*	

7. Conclusion

Clouds provide scalable processing and storage services that can be used to extract knowledge from large data repositories, as well as software platforms for developing and running data analysis environments on such services. In this article, an overview of cloud

technologies by describing the main service models (software as a service, platform as a service, and infrastructure as a service) and deployment models (public, private, or hybrid clouds) adopted by cloud providers. have been, we presented. We also described examples of cloud environments (Microsoft Azure, Amazon Web Services, OpenNebula, and OpenStack) that can be used to implement applications and frameworks for data analysis in the cloud. Finally, after identifying the main needs to be met by a distributed data analysis system, we describe as an example how Microsoft Azure components and methods can be used to meet these needs.

References

1. *R. Barga, D. Gannon, and D. Reed, "The client and the cloud: Democratizing research computing," IEEE Internet Computing, 15(1):72–75, 2021.*
2. *Li, A., Yang, X., Kandula, S., Zhang, M., 2020. CloudCmp: comparing public cloud providers. Tenth ACM SIGCOMM Conference on Internet Measurement (IMC'10), New York, USA.*
3. *Mell, P., Grance, T., 2019. The NIST Definition of Cloud Computing. NIST Special Publication 800-145.*
4. *Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D., 2018. The eucalyptus open-source cloud computing system. In: Proceedings of the Ninth IEEE/ ACM International Symposium on Cluster Computing and the Grid (CCGRID'09), Washington, USA.*
5. *Richardson, L., Ruby, S., 2015. RESTful Web Services. O'Reilly & Associates, California.*
6. *Beshkani, Mohammad Kazem and Rajaei, Zahra, 2017, Examining effective patterns in the intelligent education system based on cloud computing, The 5th National Conference on Computer Science and Engineering and Information Technology, Babol, https://civilica.com/doc/810338*
7. *Beshkani, Mohammad Kazem and Akbarpour Koumleh, Maria, 2017, Electronic Education System Architecture Based on Cloud Processing, Fifth National Conference on Computer Science and Engineering and Information Technology, Babol, https://civilica.com/doc/810337*
8. *Beshkani, Mohammad Kazem and Jameh Shoorani, Mohammad, 2018, Analysis of security features and challenges in cloud computing, 7th National Conference on Computer Science and Engineering and Information Technology, Babol, https://civilica.com/doc/913332*
9. *Mohammadian Khazineh, Benyamin and Asgari Mehrabadi, Shaghaigh and Latifi, Seyed Ahmad and Beshkani, Mohammad Kazem, 2021, Analysis of Cloud Computing Challenges, 6th International Conference on Applied Research in Computer, Electricity and Information Technology, https://civilica.com /doc/1452687.*
10. *Sotomayor, B., Montero, R.S., Llorente, I.M., Foster, I., 2015. Virtual infrastructure management in private and hybrid clouds. IEEE Internet Comput. 13, 14–22.*