

Unit 1 Questions

1. Briefly explain Data management in Enterprise system and its components and best practices for the same.

Data management in an enterprise system refers to the process of organizing, storing, protecting, and maintaining data to ensure its accuracy, availability, and reliability. It involves the use of various components and best practices to ensure that data is effectively managed throughout its lifecycle.

Components of Data Management in Enterprise Systems:

1. **Data Architecture:** This involves the design and organization of data structures, data models, and data flows within an enterprise system.
2. **Data Governance:** This involves the policies, procedures, and processes for managing and protecting data within an enterprise system. It also includes the roles and responsibilities of stakeholders in managing data.
3. **Data Quality Management:** This involves ensuring that data is accurate, complete, and consistent, and conforms to defined business rules.
4. **Master Data Management:** This involves creating and maintaining a single, authoritative source of master data that is used throughout an enterprise system.
5. **Metadata Management:** This involves capturing and managing metadata, which provides context and meaning to data.

Best Practices for Data Management in Enterprise Systems:

1. Establish data governance policies and procedures that align with the organization's overall goals and objectives.
2. Use a data catalog or inventory to identify and manage data assets.
3. Implement data quality controls to ensure that data is accurate, complete, and consistent.
4. Implement data security and privacy controls to protect sensitive data.
5. Establish a master data management program to ensure that master data is consistent and accurate across the enterprise.
6. Use metadata management to provide context and meaning to data.
7. Implement a data retention policy to ensure that data is retained for the appropriate period of time.
8. Ensure that data management practices are regularly reviewed and updated to reflect changes in business needs and regulatory requirements.

2. Describe in detail workflow in a business process and its types and advantages

Workflow in a business process refers to the sequence of tasks and activities that are performed to achieve a specific business objective. It involves the movement of information, documents, or tasks from one step to another in a systematic and structured manner. Workflows can be automated or manual, and can be categorized into several types, including sequential, parallel, and hybrid workflows.

Types of Workflows:

1. **Sequential Workflow:** In a sequential workflow, tasks and activities are completed in a linear or sequential manner, where the output of one task becomes the input of the next. This type of workflow is best suited for processes that have a clear and predefined sequence of steps.
2. **Parallel Workflow:** In a parallel workflow, multiple tasks and activities are performed simultaneously. This type of workflow is best suited for processes that require collaboration or involve multiple departments or individuals.
3. **Hybrid Workflow:** A hybrid workflow combines sequential and parallel workflows to achieve a specific business objective. This type of workflow is best suited for processes that have both sequential and parallel elements.

Advantages of Workflow in Business Processes:

1. **Increased Efficiency:** Workflow automates and streamlines processes, reducing the time and effort required to complete tasks and activities. This can lead to increased productivity and reduced costs.
2. **Improved Collaboration:** Workflows facilitate collaboration among team members and

departments, allowing for better communication and coordination.

3. **Better Visibility:** Workflows provide real-time visibility into the status of tasks and activities, enabling managers to identify bottlenecks and areas for improvement.
4. **Improved Compliance:** Workflows can ensure that processes are compliant with industry standards and regulations, reducing the risk of penalties and fines.
5. **Enhanced Customer Service:** Workflows can improve customer service by providing faster response times and more accurate information.

In summary, workflows are an essential aspect of business processes, enabling organizations to automate and streamline tasks and activities, improve collaboration, increase efficiency, and enhance customer service. The choice of workflow type depends on the nature of the process and the business objectives that need to be achieved.

3. Explain ERP in detail with help of a suitable diagram and its advantage and disadvantage

Enterprise Resource Planning (ERP) is a type of software application that integrates and manages all the core business processes and functions within an organization, such as finance, human resources, inventory management, supply chain management, and customer relationship management. The primary objective of an ERP system is to provide a centralized and real-time view of the organization's data and processes to facilitate decision-making and improve operational efficiency.

Diagram: Here is a high-level diagram that depicts the various modules and components of an ERP system:



Enterprise resource planning (ERP) refers to a type of software that organizations use to manage day-to-day business activities such as accounting, procurement, project management, risk management and compliance, and supply chain operations.

Advantages of ERP:

1. **Improved Efficiency:** An ERP system can streamline and automate routine tasks, reducing manual effort and increasing operational efficiency.
2. **Enhanced Collaboration:** ERP enables better collaboration among departments and functions by providing a unified view of data and processes, improving communication and coordination.
3. **Increased Visibility:** ERP provides real-time access to data and metrics, enabling managers to make informed decisions based on accurate information.
4. **Standardization:** ERP promotes standardization of processes and procedures across departments, reducing variability and improving consistency.
5. **Scalability:** ERP is highly scalable, allowing organizations to add new modules and functionality as needed to support growth and expansion.

Disadvantages of ERP:

1. **High Cost:** ERP systems can be expensive to implement, requiring significant investment in hardware, software, and customization.
2. **Complexity:** ERP systems can be complex and require significant effort to configure and customize to meet the organization's needs.
3. **Resistance to Change:** Employees may resist using a new system, which can impact user adoption and affect the overall success of the implementation.
4. **Integration Challenges:** Integrating an ERP system with other applications and systems can be challenging, requiring significant effort and resources.
5. **Dependence on Vendor:** Organizations may become dependent on the ERP vendor for support and upgrades, limiting flexibility and autonomy.

4. Discuss Supply Chain Management (SCM) & its types of models.

Supply Chain Management (SCM) refers to the management of the flow of goods, services, information, and finances across the entire supply chain, from the suppliers to the end customers. The primary goal of SCM is to optimize the entire supply chain to achieve maximum efficiency and profitability while minimizing costs and risks.

There are several types of SCM models, including:

1. **Push Model:** In a push model, the production and delivery of goods and services are based on anticipated demand or forecasts. This model is commonly used in industries with stable demand patterns, such as grocery stores.
2. **Pull Model:** In a pull model, the production and delivery of goods and services are based on actual demand. This model is commonly used in industries with highly variable demand patterns, such as fashion.
3. **Lean Model:** In a lean model, the focus is on reducing waste and improving efficiency throughout the supply chain. This model is based on the principles of lean manufacturing and aims to minimize inventory, reduce lead times, and improve quality.
4. **Agile Model:** In an agile model, the focus is on quickly responding to changing customer demands and market conditions. This model requires a flexible and responsive supply chain that can adapt to changes in demand and supply.
5. **Resilient Model:** In a resilient model, the focus is on mitigating risks and disruptions in the supply chain. This model requires a robust and redundant supply chain that can quickly recover from disruptions such as natural disasters or supply chain disruptions.

Advantages of SCM:

1. **Improved Efficiency:** SCM can improve the efficiency of the supply chain by reducing lead times, minimizing inventory, and improving quality.
2. **Cost Savings:** SCM can reduce costs by optimizing the use of resources and improving productivity.
3. **Improved Customer Service:** SCM can improve customer service by reducing lead times, improving product quality, and providing faster delivery times.
4. **Improved Collaboration:** SCM can improve collaboration among suppliers, manufacturers, distributors, and customers, improving communication and coordination.
5. **Reduced Risk:** SCM can reduce the risk of supply chain disruptions and improve the ability to respond to unforeseen events such as natural disasters or market fluctuations.

5. Elaborate Customer Relationship Management (CRM) & Components and types.

Customer Relationship Management (CRM) is a business strategy that focuses on managing interactions with customers to improve customer satisfaction, loyalty, and retention. The primary goal of CRM is to create long-term relationships with customers and increase their lifetime value.

Components of CRM:

1. **Customer Data Management:** CRM requires a robust and centralized customer database that stores all customer-related data, including personal information, transaction history, purchase behavior, and contact details.
2. **Sales Automation:** CRM systems can automate sales-related tasks, including lead generation, lead qualification, opportunity tracking, and sales forecasting.
3. **Marketing Automation:** CRM systems can automate marketing-related tasks, including email marketing, social media marketing, and lead nurturing.
4. **Customer Service and Support:** CRM systems can manage customer inquiries, complaints, and support tickets, providing a consistent and high-quality customer experience.
5. **Analytics and Reporting:** CRM systems provide analytics and reporting tools that enable organizations to measure the effectiveness of their customer engagement strategies, identify trends and patterns, and make data-driven decisions.

Types of CRM:

1. **Operational CRM:** Operational CRM focuses on improving customer-facing processes and optimizing customer interactions, such as sales, marketing, and customer service. Operational CRM aims to improve efficiency, effectiveness, and customer satisfaction.
2. **Analytical CRM:** Analytical CRM focuses on analyzing customer data to gain insights into customer behavior, preferences, and needs. Analytical CRM uses data mining, predictive modeling, and other analytical techniques to identify patterns and trends and provide actionable insights for improving customer engagement strategies.
3. **Collaborative CRM:** Collaborative CRM focuses on improving collaboration and communication among internal teams and external partners to improve customer interactions. Collaborative CRM aims to provide a seamless and consistent customer experience across all touchpoints.

Advantages of CRM:

1. **Improved Customer Experience:** CRM systems can provide a personalized and seamless customer experience across all touchpoints, improving customer satisfaction and loyalty.
2. **Increased Sales:** CRM systems can improve sales effectiveness by automating sales-related tasks, tracking customer interactions, and providing insights into customer needs and preferences.
3. **Improved Marketing ROI:** CRM systems can improve marketing ROI by automating marketing-related tasks, tracking customer behavior, and providing insights into the effectiveness of marketing campaigns.
4. **Improved Customer Retention:** CRM systems can improve customer retention by providing a high-quality customer experience, addressing customer needs and concerns promptly, and providing personalized offers and promotions.
5. **Improved Collaboration:** CRM systems can improve collaboration and communication among internal teams and external partners, improving the efficiency and effectiveness of customer-facing processes.

6. Illustrate Product Life Cycle & its stages along with its advantages and limitations

Product life cycle (PLC) is the progression of a product through its lifespan, from its introduction to the market until it is eventually withdrawn. The product life cycle is typically represented by a curve with four stages: Introduction, Growth, Maturity, and Decline.

1. **Introduction Stage:** In this stage, the product is introduced to the market. Sales are low, and the company may incur losses due to the high costs of developing and launching the product.

Advertising and promotional activities are focused on creating awareness and generating interest among potential customers.

2. **Growth Stage:** In this stage, sales increase rapidly, and the product gains acceptance in the market. Competition may increase, and the company may need to invest in additional production capacity to meet the growing demand. Advertising and promotional activities are focused on building brand loyalty and increasing market share.
3. **Maturity Stage:** In this stage, sales growth slows down, and the product reaches its peak. The market becomes saturated, and competition becomes intense. The company may need to reduce prices, increase promotions, or introduce product improvements to maintain market share.
4. **Decline Stage:** In this stage, sales decline, and the product is eventually withdrawn from the market. The decline may be due to changes in customer preferences, technological advances, or the emergence of new products that offer better value.

Advantages of Product Life Cycle:

1. Helps companies to plan and allocate resources based on the stage of the product.
2. Provides insights into the sales and profitability of the product.
3. Allows companies to identify potential problems and develop strategies to overcome them.
4. Helps companies to understand the dynamics of the market and competition.

Limitations of Product Life Cycle:

1. Assumes that products follow a predictable and uniform life cycle, which may not be true in all cases.
2. Does not take into account the impact of external factors such as changes in technology, regulations, or economic conditions.
3. May not accurately reflect the sales and profitability of the product due to variations in product quality, price, and distribution.
4. May not consider the impact of marketing activities, such as advertising and promotion, on the life cycle of the product.

7. Briefly explain Human Resource Management (HRM)

Human Resource Management (HRM) refers to the management of human capital within an organization. It involves recruiting, hiring, training, and managing employees to achieve organizational goals. HRM also includes developing and implementing policies and procedures that ensure the fair treatment of employees and compliance with legal requirements.

The primary functions of HRM include:

1. **Recruitment and selection:** Attracting and selecting the right people for the job.
2. **Training and development:** Providing employees with the necessary knowledge and skills to perform their job effectively.
3. **Performance management:** Evaluating and managing employee performance to achieve organizational goals.
4. **Compensation and benefits:** Determining and administering employee compensation and benefits packages.
5. **Employee relations:** Maintaining positive relationships between employees and management and addressing workplace issues and conflicts.
6. **Compliance with legal requirements:** Ensuring that the organization complies with employment laws and regulations.

HRM is essential for the success of any organization, as it helps to ensure that the right people are in the right positions and that they are equipped with the necessary skills and knowledge to perform their jobs effectively. It also helps to create a positive workplace culture that fosters employee engagement, retention, and productivity.

8. What are General Ledger Systems?

A General Ledger System (GLS) is a financial accounting software system used by organizations to maintain and manage their financial transactions. It provides a centralized repository of all financial transactions, which can be used to generate financial reports and analyses financial performance. The primary function of a GLS is to record and classify financial transactions into specific categories such as assets, liabilities, revenues, expenses, and equity. The GLS also provides a mechanism for creating and managing journal entries, which are used to adjust the financial records for any errors or omissions.

The key features of a GLS include:

1. Chart of Accounts: A predefined list of accounts used to classify financial transactions.
2. Journal Entry: A mechanism to record financial transactions and adjust the financial records.
3. Trial Balance: A summary of all the account balances to ensure that the total debits and credits are equal.
4. Financial Reporting: A mechanism to generate financial statements such as income statements, balance sheets, and cash flow statements.
5. Audit Trail: A record of all financial transactions to track any changes made to the financial records.

The benefits of using a GLS include:

1. Increased accuracy and efficiency: A GLS provide a centralized repository for financial transactions, which reduces the likelihood of errors and makes the process more efficient.
2. Better financial reporting: A GLS provides a mechanism for generating accurate and timely financial reports, which are essential for decision-making.
3. Improved compliance: A GLS ensures compliance with accounting standards and regulations, reducing the risk of penalties and legal issues.
4. Enhanced decision-making: A GLS provides accurate and timely financial information, which can be used to make informed decisions about the organization's financial performance.

Unit 2 Questions

1. Explain Distributivity and how to secure distributed enterprise?

Distributivity refers to the property of a system or process that is distributed across multiple locations or nodes. In a distributed system, the processing power, storage, and other resources are spread across multiple nodes, which enables better scalability, fault tolerance, and performance.

However, securing a distributed enterprise can be challenging due to the following reasons:

1. Increased attack surface: A distributed enterprise has a larger attack surface as it involves multiple nodes and communication channels, making it vulnerable to various cyber threats such as hacking, malware, and phishing attacks.
2. Complex network topology: A distributed enterprise involves a complex network topology, which can be difficult to manage and secure. This complexity makes it challenging to ensure proper authentication, access control, and data protection.
3. Data synchronization and consistency: In a distributed enterprise, data needs to be synchronized across multiple nodes to ensure consistency. However, this synchronization can be difficult to achieve and can result in data inconsistencies, which can compromise data integrity and security.

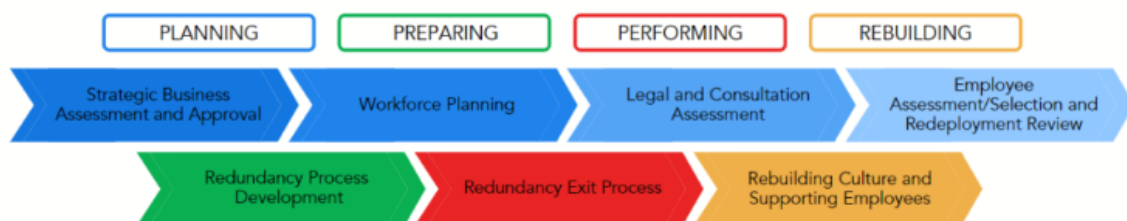
To secure a distributed enterprise, the following best practices can be followed:

1. Network segmentation: Segregate the network into smaller, more manageable segments to reduce the attack surface and limit the impact of any security breaches.
2. Authentication and access control: Implement strong authentication and access control mechanisms such as multi-factor authentication, role-based access control, and least privilege access to ensure that only authorized users can access sensitive data and resources.

3. Data encryption: Use strong encryption algorithms to protect sensitive data and ensure that data is protected both at rest and in transit.
4. Continuous monitoring and threat detection: Implement continuous monitoring and threat detection mechanisms to detect and respond to security threats in real-time.
5. Disaster recovery and business continuity planning: Develop and implement disaster recovery and business continuity plans to ensure that the organization can recover quickly in case of any security breaches or other disruptions.

2. What is a redundancy process? What is the best redundancy process to follow?

A redundancy process is a process where an organization reduces its workforce by terminating the employment of one or more employees due to reasons such as financial constraints, changes in business strategy, or restructuring. Redundancy can be a difficult process for both the employees and the organization, and it is essential to follow a fair and transparent process to minimize the impact on the affected employees and maintain the organization's reputation.



Strategic Business Assessment and Approval

1. Inspect the current state of your business / organisation.
2. Develop a map of your business goals and strategy to get a clear vantage point.
3. When working for a larger organisation, investigate if you are eligible to run the redundancy. If you belong to a smaller business, understand the impacts on your budget.
4. In some cases, running the redundancy requires other corporate members.

Workforce Planning

- Analyse your talent against your current goals.
- Consider the future and the skills and experience you have to meet your future strategy.
- Identify the gaps in your workforce and assess the external market.
- Determine the solution that suits your needs and understand if redundancies are required.

Legal and Consultation Assessment

- Assess legal requirements and ensure you know your obligations under the law.
- Assess Industrial Relations/Employee Relations Risks and Union consultation or collaboration requirements under any enterprise agreements.
- Ensure that the people involved sign confidentiality statements and understand the implications of breaching these.

Employee Assessment/Selection and Redeployment Review

- Determine if you even need to run a selection process. If your redundancy program involves reducing the number of people performing the same roles, then the answer will be yes.
- Carry out a sound assessment process—include evidence requirements, relevant training for leaders and assessment procedures.

Redundancy Process Development

- Develop a redundancy and exit process, considering the most dignified approach for impacted employees.
- Develop templates, tools, scripts and training that may be required to ensure consistency and professionalism of your program.
- Using scripts to support all redundancy meetings will ensure conversations stay factual, objective and free of bias.

Redundancy Exit Process

Distribute communications to all relevant stakeholders in line with your communication plan. Consider the following:

- Appropriate Dates and Venue
- Communication to Employees
- Structure of Meetings
- Employee Support
- Exit Checklist
- Government Notification

Rebuilding Culture and Supporting Employees

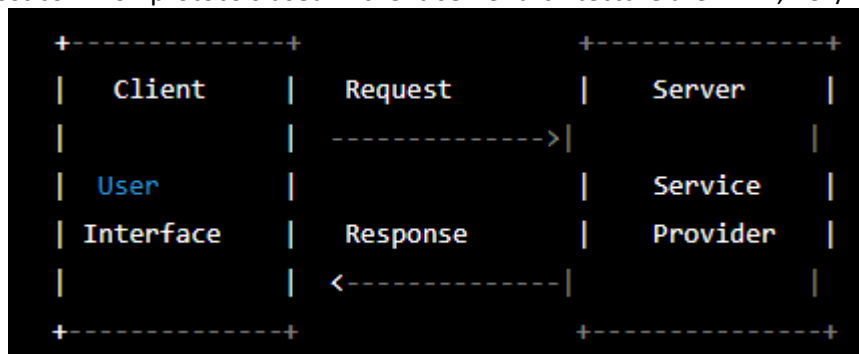
- Communicate to the workforce about the end of the process
- Acknowledge and honour those that left and create opportunities for people to talk
- Monitor individuals to ensure mental health is intact—be aware of survival guilt
- Coordinate employee assistance program (EAP) or mental health support. Initiate engagement activities over the following weeks

3. Elaborate client server architecture with suitable diagram and its type

Client-server architecture is a computing model in which clients request services or resources from servers, which provide the requested services or resources. This architecture involves a distributed computing model where the processing power and resources are split between the client and server. The client-server architecture consists of the following components:

1. Client: A client is a device or software application that requests services or resources from the server. Clients can be desktop applications, mobile applications, web browsers, or any other device that can connect to a network.
2. Server: A server is a device or software application that provides services or resources to clients. Servers can be web servers, file servers, database servers, or any other device that can respond to client requests.
3. Network: The network connects clients and servers and enables communication between them. Networks can be local area networks (LANs), wide area networks (WANs), or the internet.

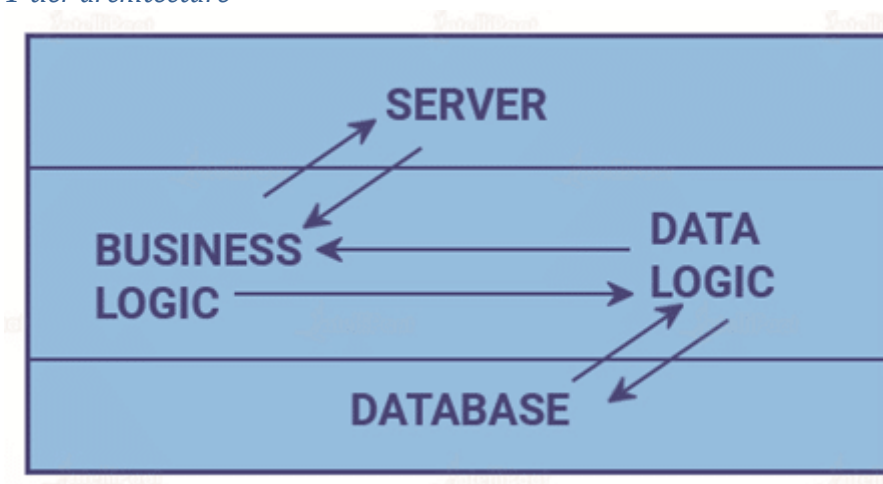
4. Protocol: A protocol is a set of rules that governs communication between clients and servers. The most common protocols used in client-server architecture are HTTP, TCP/IP, and SMTP.



Types of client server architecture

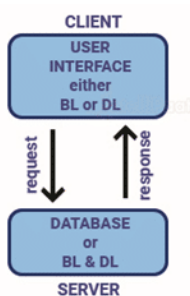
The functionality of client server architecture is in various tiers.

1-tier architecture



In this category of client server architecture, the architecture contains all kinds of settings, such as configuration setting and marketing logic, on a single device. While the diversity of services offered by 1-tier architecture makes it one of the reliable sources, handling such an architecture is difficult. This is primarily due to the data variance. It often results in replication of work. 1-tier architecture consists of several layers, such as presentation layer, business layer, and data layer, that are combined with the help of a unique software package. The data present in this layer is usually stored in local systems or on a shared drive.

2-tier architecture

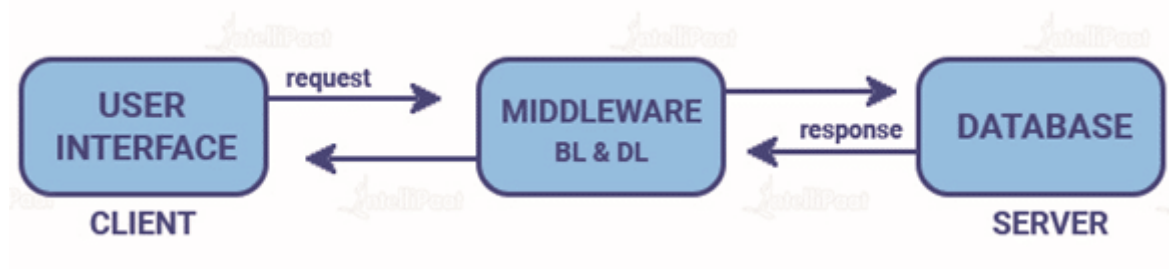


his architecture has the best environment. In this architecture, the user interface is stored on the client's side and the database is stored on the server, while database logic and business logic is maintained either on the client's side or on the server's side.

The 2-tier architecture is faster in comparison to the 1-tier architecture; this is because the 2-tier architecture does not have any intermediary between the client and the server. It is often

utilized to avoid confusion between clients. One of the popular examples of 2-tier architecture is the online ticket reservation system.

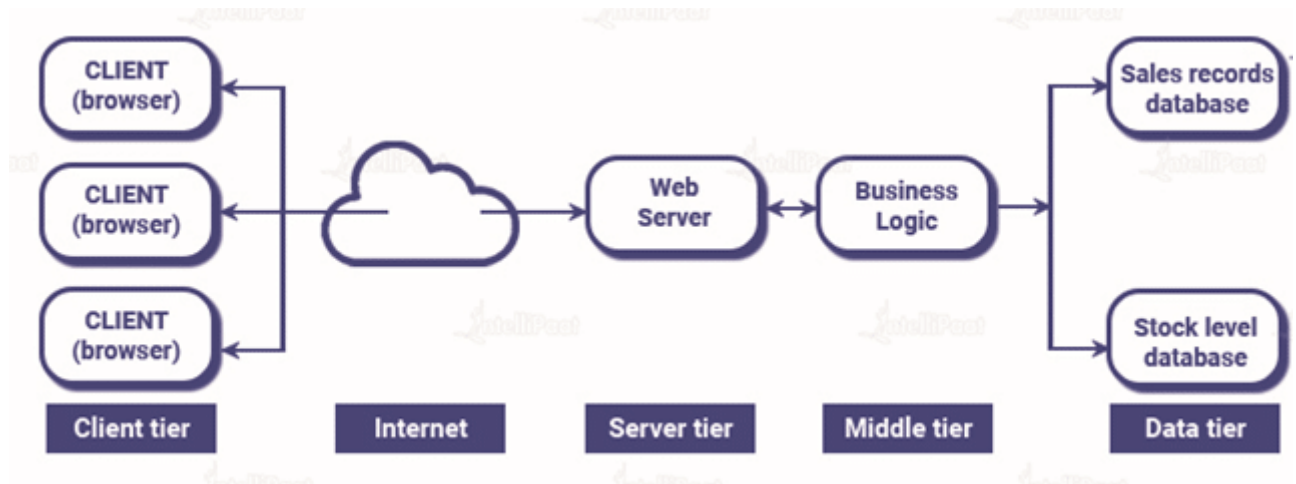
3-tier architecture



Unlike 2-tier architecture that has no intermediary, in 3-tier client server architecture, a middleware lies between the client and the server. If the client places a request to fetch specific information from the server, the request will first be received by the middleware. It will then be dispatched to the server for further actions. The same pattern will be followed when the server sends a response to the client. The framework of 3-tier architecture is categorized into three main layers, presentation layer, application layer, and database tier.

All three layers are controlled at different ends. While the presentation layer is controlled at the client's device, the middleware and the server handle the application layer and the database tier respectively. Due to the presence of a third layer that provides data control, 3-tier architecture is more secure, has invisible database structure, and provides data integrity.

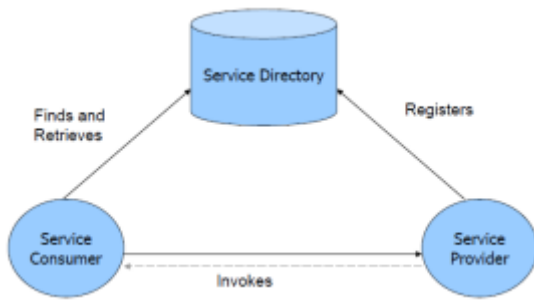
N-tier architecture



N-tier architecture is also called multi-tier architecture. It is the scaled form of the other three types of architecture. This architecture has a provision for locating each function as an isolated layer that includes presentation, application processing, and management of data functionalities.

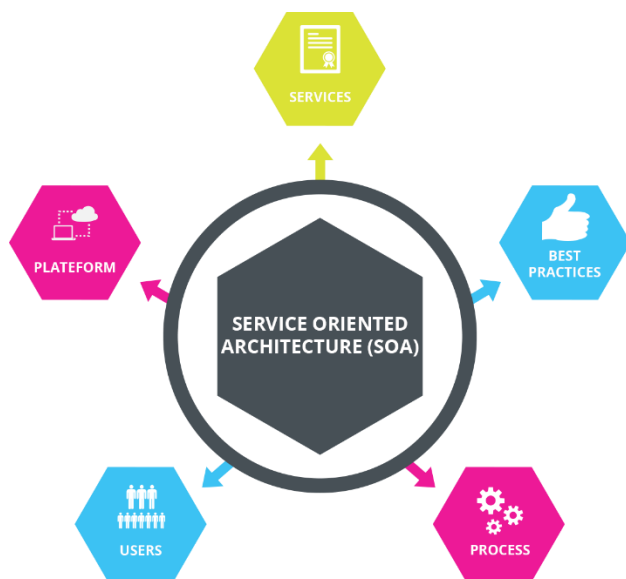
4. Illustrate Service Oriented Architecture with help of suitable diagram

Service Oriented Architecture (SOA) is an architectural style that involves the creation of reusable services that can be accessed by different applications and systems. SOA enables businesses to build flexible and scalable systems that can adapt to changing business requirements.



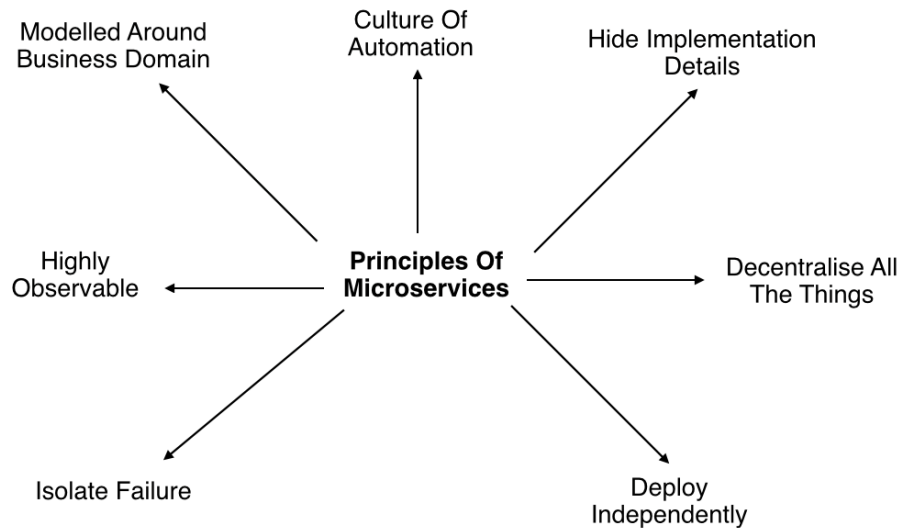
A SOA has three major parts; **service provider, service consumer, and service directory**. **Service providers** are the parties who build service and make available service. **Service consumers** are the clients who consume services. **Service directory** is the place where service providers register the services and consumer search for services. Service directory provide following services: 1. Scalability of services; can add services incrementally. 2. Decouples consumers from providers. 3. Allows for hot updates of services. 4. Provides a look-up service for consumers. 5. Allows consumers to choose between providers at runtime rather than hard-coding a single provider

There are three roles in each of the Service-Oriented Architecture building blocks: service provider; service broker, service registry, service repository; and service requester/consumer.



5. Describe Micro- service principles and architecture.

Microservice Principle



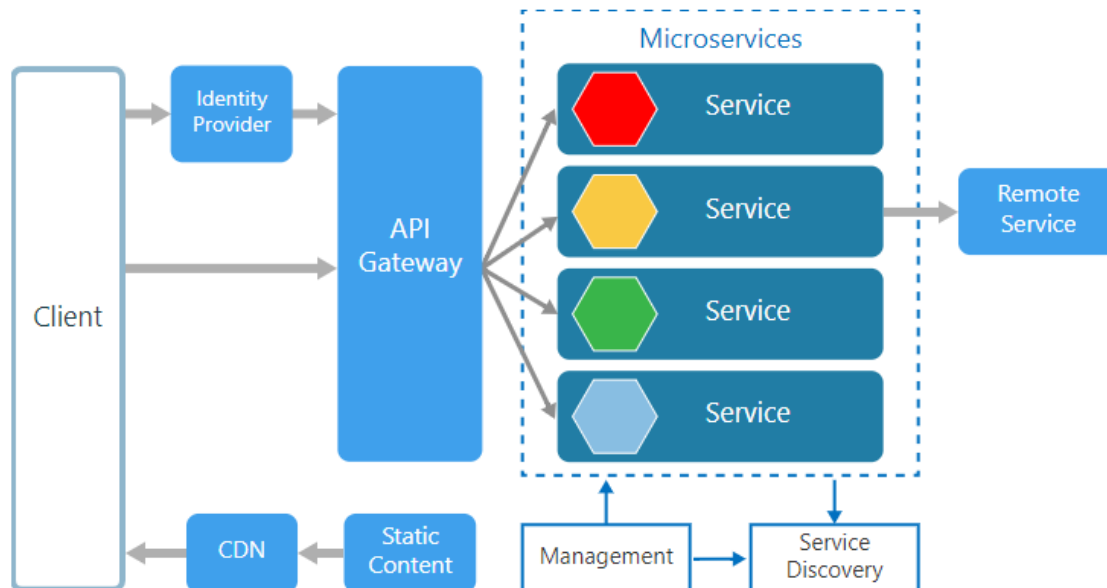
Modelled around business domain: Micro-services architecture lets us separate system capability into different domains. Each domain will focus on one thing and its associated logic and can easily migrate independently to the next version, and also scale independently according to requirement.

- **Culture of Automation:** As we are building, testing, deploying, and monitoring each service separately and there is an increase in the number of deployment units compared to monolithic architecture we should follow the culture of automation by designing it for continuous integration and continuous delivery.
The smaller and compact codebases and their defined scope are generally turned out to quicker deployments, which also allow us to start to explore the benefits of Continuous Deployment and Continuous integration seamlessly.
- **Hide implementation details:** Micro-services should be architected in such a way that they won't expose the internal details; neither technical implementation nor the business rules that drive it. This will reduce the coupling and help to do changes and improvements without affecting the overall architecture.
- **Decentralization:** In traditional monolithic implementations, the software is designed to use a single database with different tables whereas micro-services are designed in such a way to manage their own database.
- **Deploy Independently:** To enjoy the complete benefits of the architecture, Micro-services should be independently deployable. If you are failing to do so, check for any coupling in the application and solve it.
- **Failure Isolation:** The impact of a failure is less in Micro-services architecture compares to the monolithic type as it will only affect that particular service and its association while other services can keep running. The associated services should handle such scenarios when the dependent is unresponsive or slow.
The larger or the enterprise applications may remain unaffected mostly by the failure of a single module and due to that, other parts of the application are running concurrently which enhances the availability of the feature to the customers most of the time.
- **Highly Observable:** The services should collect as much information to analyze what is

happening within each of them like log events and stats.

Microservice Architecture

An architectural style that structures an application as a collection of small self-contained processes, modelled around a business capability. They don't share the data structure and will be communicating through APIs. While in a monolithic application all the components are in a single module, in Micro-services we can see all the components are divided into a separate module and communication happens with each other with the help of APIs. In Micro-services Architecture the data is federated where each Micro-services is responsible for its own data model and data.

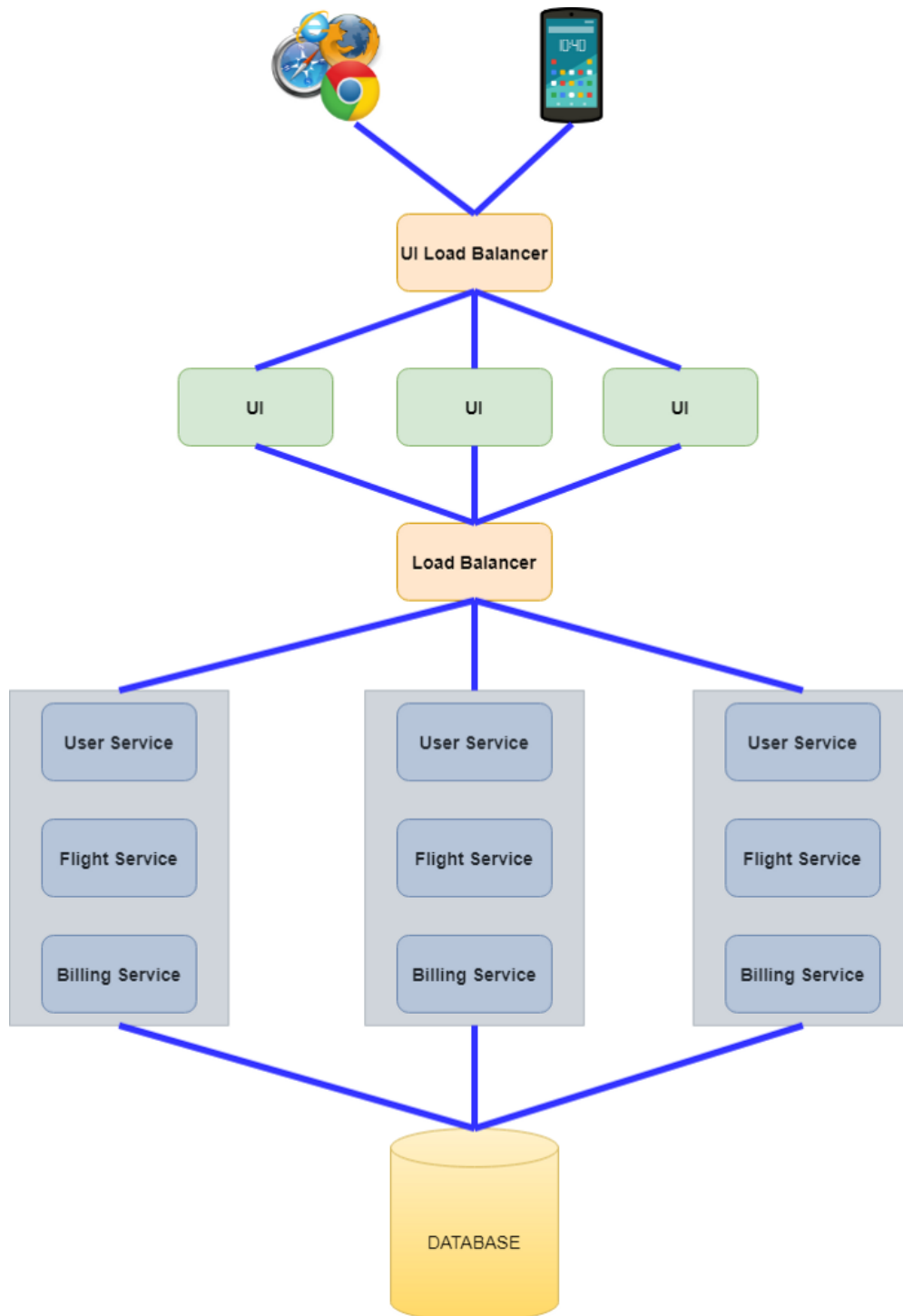


Being small in size, independent, and loosely coupled each service can be developed and deployed independently by a small team as each service is having its own codebase. Data and state persistence should be taken with each service as it lacks a separate data layer to handle it. The services only communicate with well-defined APIs hiding each service's internal implementation from the other. Each service can use a different technology stack, language, libraries, or frameworks.

- **Management:** The Management takes care of the placement of services on nodes, checking for failures, and rebalancing services across nodes in case of any failures.
- **Service Discovery:** Maintains a list of services and the nodes where each service is located, and also enables the service to look up to find the endpoint for a particular service.
- **API Gateway:** The entry point for clients where all the calls from the client will be taken, analyze, and forward to appropriate services. In case some calls are needed from multiple services API Gateway will aggregate and will return the aggregated result.

6. Sketch and elaborate monolithic architecture and state its advantages and disadvantages.

Monolithic Architecture is a traditional way of building applications. This software architecture principle has both advantages and disadvantages. On the one hand, it can bring delight. On the other hand, it can cause disappointment. Let's review its place in software architecture.



Typical Monolithic Architecture is presented in the picture above. The key characteristic of it is the fact that everything (*User Service*, *Flight Service*, and *Billing Service*) is located inside in a single deliverable (a jar file in case of Spring Boot). These services are tightly coupled (despite the fact that they have totally different functions) and use the same database.

Advantages of Monolithic Architecture

- **Simplicity of development.** The monolithic approach is a standard way of building applications. No additional knowledge is required. All source code is located in one place which can be quickly understood.
- **Simplicity of debugging.** The debugging process is simple because all code is located in one place. You can easily follow the flow of a request and find an issue.
- **Simplicity of testing.** You test only one service without any dependencies. Everything is usually clear.
- **Simplicity of deployment.** Only one deployment unit (e.g. *jar* file) should be deployed. There are no dependencies. In cases when UI is packaged with backend code you do not have any breaking changes. Everything exists and changes in one place.
- **Simplicity of application evolution.** Basically, the application does not have any limitation from a business logic perspective. If you need some data for new feature, it is already there.
- **Cross-cutting concerns and customizations are used only once.** You should take care of cross-cutting concerns only once. For instance, security, logging, exception handling, monitoring, choosing and setting up tomcat parameters, setup of data source connection pool, etc.
- **Simplicity in onboarding new team members.** The source code is located in one place. New team members can easily debug some functional flow and to get familiar with the application.
- **Low cost in the early stages of the application.** All source code is located in one place, packaged in a single deployment unit, and deployed. What can be easier? There is no overhead neither in infrastructure cost nor development cost.

Disadvantages of Monolithic Architecture

- **Slow speed of development.** The simplest disadvantage relates to CI/CD pipeline. Imagine the monolith that contains a lot of services. Each service in this monolith is covered with tests that are executed for each Pull Request. Even for a small change in a source code, you should wait a lot of time (e.g. 1 hour) for your pipeline to succeed
- **High code coupling.** Of course, you can keep a clear service structure inside your repository. However, as practice shows, eventually, you will end up with a spaghetti code in at least a few places. As a result, the system becomes harder to understand especially for new team members.
- **Code ownership cannot be used.** The system is growing. The logical step is to split responsibilities between several teams. E.g. one team can work on *Flight Service*, another — for *Billing Service*. However, there are no boundaries between those services. One team can affect another.
- **Testing becomes harder.** Even a small change can negatively affect the system. As a result, the regression for full monolithic service is required.

- **Performance issues.** Potentially, you can scale the whole monolithic service in cases of performance issues. But what to do with the database? The single database is used for all services. You can start to optimize your database queries or use read replicas. However, there is a limit to this type of optimization.
- **The cost of infrastructure.** In cases of performance issues, you should scale the whole monolithic service. It brings additional costs for application operability.
- **Lack of flexibility.** Using Monolithic Architecture you are tight to the technologies that are used inside your monolith. You cannot use other tools even if they are better for the problem at hand.
- **Problems with deployment.** Even a small change requires the redeployment of the whole monolith.

7. Distinguish between Monolithic vs SOA vs Microservices

Feature	Monolith	SOA	Microservices
Architecture	A single, self-contained application that includes all of the application's functionality.	A distributed architecture that uses standardized protocols to enable communication between services.	A distributed architecture that breaks an application down into small, independent services that communicate with each other using lightweight protocols.
Advantages	<ul style="list-style-type: none"> - Simple to develop and deploy. - Easy to maintain. 	<ul style="list-style-type: none"> - Reusability, as services can be shared between different applications and systems. - Flexibility, as services can be modified and updated without impacting other parts of the system. - Interoperability, as services can be accessed using different technologies and platforms. - Scalability, as services can be scaled independently of each other. 	<ul style="list-style-type: none"> - Scalability, as each service can be scaled independently of each other. - Flexibility, as each service can be modified and updated without impacting other services. - Resilience, as failures in one service do not necessarily impact the entire application. - Easier to update and deploy, as changes can be made to individual services without re-deploying the entire application.
Disadvantages	<ul style="list-style-type: none"> - Limited scalability, as the entire application must be scaled as a single unit. - Difficult to update and deploy, as any changes require re-deploying the entire application. 	<ul style="list-style-type: none"> - Complex to design and implement, as the architecture requires careful planning and coordination. - Potential for service dependency issues, as changes to one service can impact other services that depend on it. 	<ul style="list-style-type: none"> - Complexity of managing multiple services, as the architecture requires careful coordination and monitoring. - Potential for service dependency issues, as changes to one service can impact other services that depend on it - Increased overhead, as there is additional network communication overhead between services.

8. Define cloud computing and explain its architecture

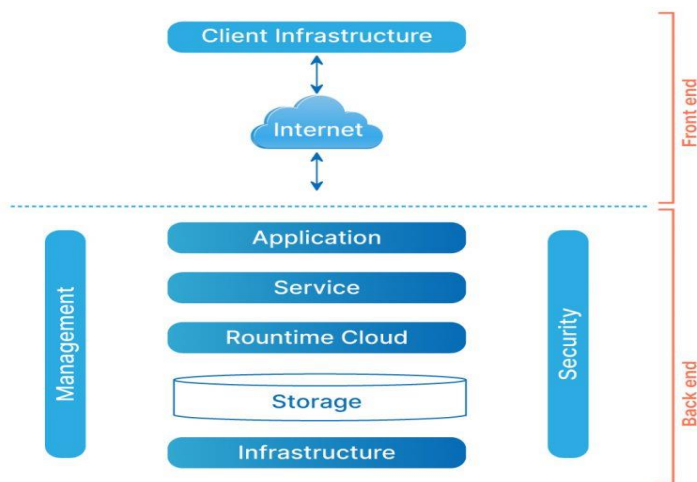
Cloud computing refers to the delivery of computing resources, including software, storage, and processing power, over the internet. Rather than maintaining their own computing infrastructure, users can access these resources on-demand from a cloud service provider, typically paying for only what they use.

Cloud architecture consists of two major parts:

- Front End
- Back End

Here's the cloud computing architecture diagram:

ARCHITECTURE OF CLOUD COMPUTING

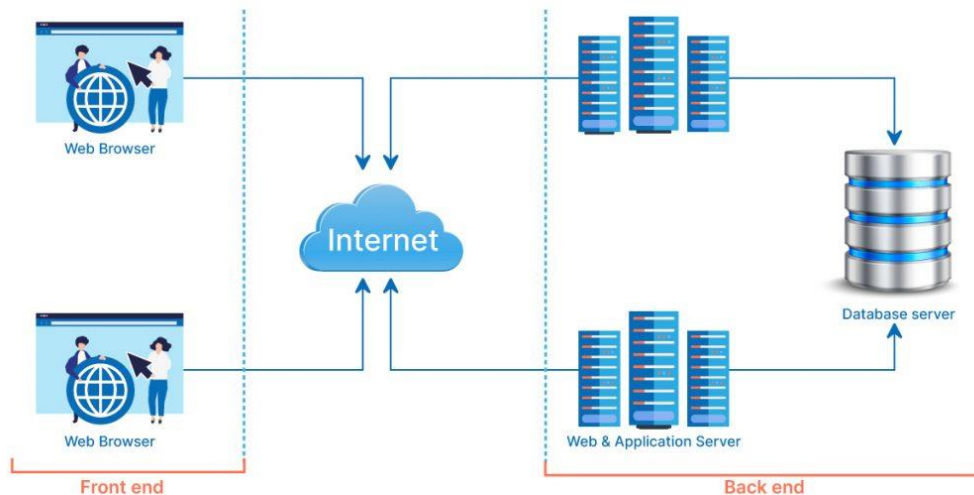


What is the Front End of cloud architecture?

Everything with which the end user interacts is part of the front-end infrastructure. The user interface is the result of integrating various sub-components, such as browsers, tablets, mobile devices, etc. With the help of the front end, the end user can connect to the cloud computing infrastructure. **In short, "the front end is the end that the client interacts with."**

What is the Back End of cloud architecture?

The back end is everything the user does not usually see and everything that processes the data. The service provider uses the back end to manage all the resources required to provide cloud computing services, such as data storage, security mechanisms, virtual machines, deploying models, servers, traffic control mechanisms, and so on. **In short, "the back end is the end that service provider interacts with."**



Components of cloud computing architecture

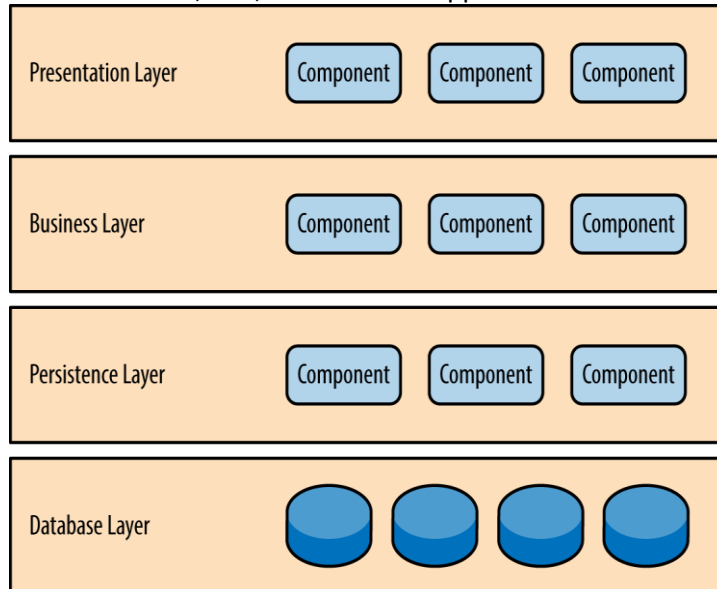
There are various components of cloud architecture. Some of those components are:

- **Client infrastructure:** The client infrastructure component is a frontend component that provides a graphical user interface (GUI) for users to interact with the cloud.
- **Application:** An application is any platform, such as an app or software, provided by a company through which clients can access the cloud.
- **Service:** A cloud service manages the type of service a client uses based on his needs. There are three types of services: SaaS (software as a service), PaaS (platform as a service), and IaaS (infrastructure as a service).
- **Runtime cloud:** The runtime cloud offers virtual machine implementation and runtime environment.
- **Storage:** The cloud computing storage component provides storage space in the cloud for managing and storing data. Cloud storage is of three types: public cloud, private cloud, and hybrid cloud.
- **Infrastructure:** The infrastructure component provides services on three levels: the host, the application, and the network. It includes the software and hardware components required to support the cloud computing model, such as storage network devices, servers, and other storage resources.
- **Management:** The management component is in charge of managing backend components such as storage services, applications, runtime cloud infrastructure, and security issues, as well as establishing coordination.
- **Security:** Security is the backend component of cloud computing that ensures data security in the cloud.
- **Internet:** The internet is the medium through which the frontend and backend components communicate and interact.

Unit 3 Questions

1. Illustrate Layer Architecture & each layer in it. Support your with and example.

Layered architecture, also known as the n-tier architecture, is a software design pattern that separates the different components of an application into layers. Each layer is responsible for a specific set of functions and communicates with the layers above and below it using well-defined interfaces. This approach makes it easier to maintain, test, and scale the application.



- Presentation layer would be responsible for handling all user interface and browser communication logic.
- Business layer would be responsible for executing specific business rules associated with the request.
- The persistence layer, also called the data access layer, acts as a protective layer. It contains the code that's necessary to access the database layer. This layer also holds the set of codes that allow you to manipulate various aspects of the database.
- Database layer is responsible for handling data, databases.

For example, when a user submits a form in the web application, **the presentation layer** would receive the form data and pass it to the business layer. **The business layer** would validate the data, perform any necessary business logic, and coordinate with the persistence layer to store or retrieve data. The business layer would then communicate with the presentation layer to display the results of the request to the user.

When a user requests data from the web application, such as a list of products or a customer record, the presentation layer would receive the request and pass it to the business layer. The business layer would coordinate with **the persistence layer** to retrieve the requested data from the database. The business layer would then communicate with the presentation layer to display the data to the user.

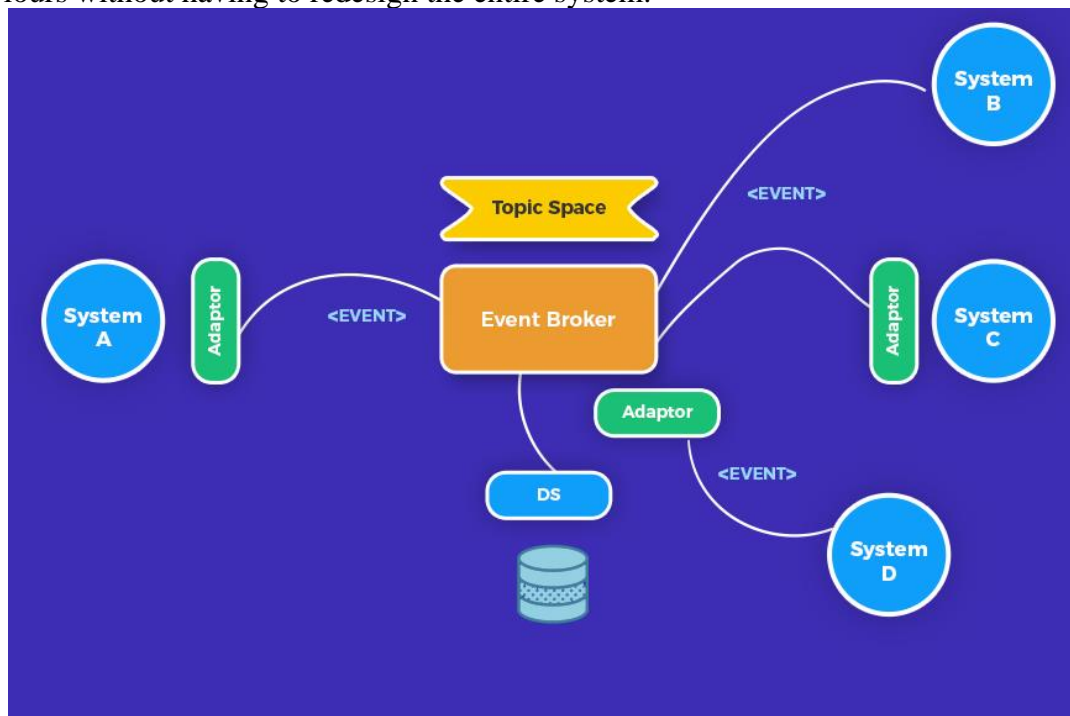
2. Elaborate Event Driven Architecture with suitable diagram.

Event-driven architecture models your business systems as a flow of events — when an important business event happens, your systems are alerted to that change of state. A simple example could be a customer changing their address: once that state change is registered, your billing systems get notified of the new address. This is in stark contrast to traditional request-based architectures.

A major benefit of this architectural pattern is that it is both scalable and relatively easy to change. EDA's inherently loosely coupled nature means that it's relatively easy to make changes in one particular part of your systems, without breaking anything else.

A well-designed EDA will be based on events that are meaningful to the business. The events could be triggered by user activity, external inputs, such as sensor activity, or outputs from an analytics system. What's important is the way you define those events, so that you're capturing something important to your organization.

By basing your designs on these triggering events, you gain flexibility; you're able to add new behaviours without having to redesign the entire system.



3. Discuss the advantages and disadvantages of SOA and give your opinion

Advantages of SOA:

1. **Modularity:** SOA allows for the creation of small, independent services that can be reused across multiple applications, reducing the need to develop and maintain duplicate functionality.
2. **Flexibility:** SOA's loosely coupled architecture allows services to be added, removed, and modified without impacting other services, making it easier to adapt to changing business requirements.
3. **Scalability:** SOA allows for the creation of scalable services that can handle large volumes of requests, ensuring that the application remains performant even under heavy loads.
4. **Interoperability:** SOA allows services to be developed using different programming languages, platforms, and technologies, enabling them to communicate with each other seamlessly.

Disadvantages of SOA:

1. **Complexity:** SOA can be complex to design and implement due to its distributed nature, making it difficult to manage and troubleshoot.
2. **Overhead:** The additional layers of abstraction and communication involved in SOA can introduce overhead, slowing down the performance of the application.

3. Security: SOA's distributed nature can make it more difficult to ensure the security of the application, requiring additional measures to protect against attacks.
4. Integration: Integration with existing systems can be challenging, requiring additional effort to ensure that data is exchanged correctly and consistently.

My opinion:

SOA can be a powerful approach to developing complex, distributed systems that need to be highly scalable and flexible. However, it requires careful planning and design to be successful, and the added complexity and overhead can make it less suitable for smaller, less complex systems. In my opinion, SOA should be considered on a case-by-case basis, with careful consideration given to the specific requirements and constraints of the application being developed.

4. Briefly explain Plug-in Architecture support your answer with a suitable diagram

A plug-in architecture is a software design pattern that allows applications to be extended with new functionality by loading and executing external code modules, called plug-ins or extensions. In this architecture, the application provides a framework that defines a set of interfaces or extension points, which the plug-ins can implement or extend to provide additional features.

Here is a diagram that illustrates the plug-in architecture:



Advantages of Plug-in Architecture:

1. Extensibility: Plug-in architecture allows the application to be easily extended with new functionality without modifying the core code.
2. Modularization: Plug-in architecture promotes the modularization of code, making it easier to maintain and debug.
3. Flexibility: Plug-in architecture provides greater flexibility, as users can choose which plug-ins to install and use.
4. Reusability: Plug-in architecture allows for the reuse of code across multiple applications.

Disadvantages of Plug-in Architecture:

1. Complexity: Plug-in architecture can introduce additional complexity, as the application must manage the loading and unloading of plug-ins.
2. Compatibility: Plug-in architecture can introduce compatibility issues, as different plug-ins may have conflicting dependencies or requirements.
3. Security: Plug-in architecture can introduce security risks, as plug-ins may be provided by third parties and may not be trustworthy.

5. Describe mapping of database and steps for the same

Data mapping is crucial to the success of many data processes. One misstep in data mapping can ripple throughout your organization, leading to replicated errors, and ultimately, to inaccurate analysis.

Step 1: Define — Define the data to be moved, including the tables, the fields within each table, and the format of the field after it's moved. For data integrations, the frequency of data transfer is also defined.

Step 2: Map the Data — Match source fields to destination fields.

Step 3: Transformation — If a field requires transformation, the transformation formula or rule is coded.

Step 4: Test — Using a test system and sample data from the source, run the transfer to see how it works and make adjustments as necessary.

Step 5: Deploy — Once it's determined that the data transformation is working as planned, schedule a migration or integration go-live event.

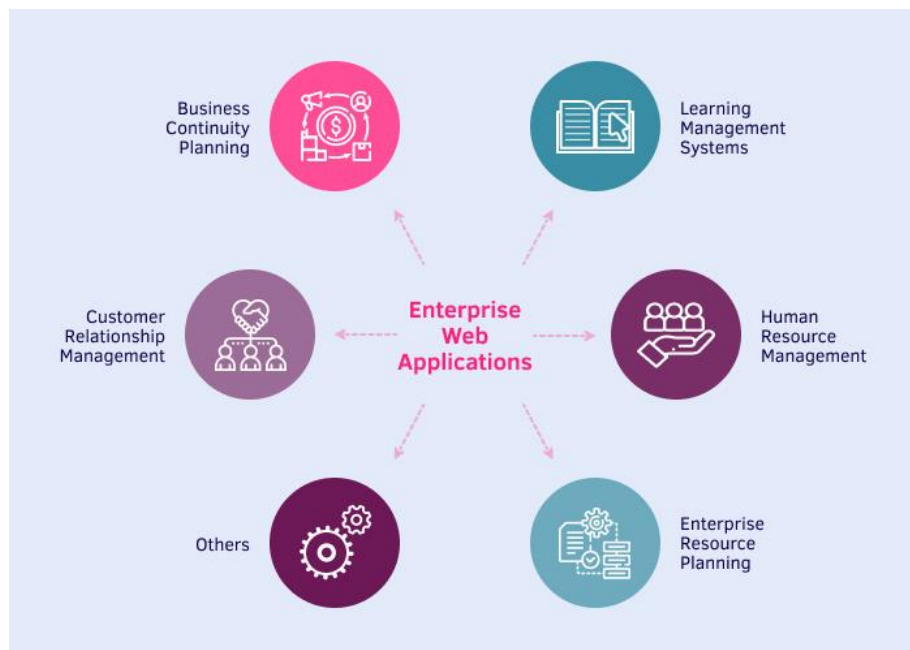
Step 6: Maintain and Update — For ongoing data integration, the data map is a living entity that will require updates and changes as new data sources are added, as data sources change, or as requirements at the destination change.

6. Illustrate Web Presentation

Enterprise Web Applications allow you to manage and record the internal and external operations and processes of your company or organization. The main advantages are the important cost savings and time savings, by helping the modern company in their digital transformation path.

Enterprise web application development is the process of programming an application for a large-scale business that is aimed at in-house usage and helps to be in control over all operational functions. This software is different from off-the-shelf web technologies because it's designed for the needs of a specific company, not its clients.

Enterprise web custom solutions can be corporate learning management systems (LMS), human resource management platforms (HRM), enterprise resource planning apps (ERP), customer relationship management software (CRM), business continuity planning programs (BCP), and many others.



3 Different Types of Enterprise Application Features

Enterprise software differs from traditional apps because it must have certain features that help the whole team be informed of all internal operations.

There are three types of such specifications:

1. Basic features

– Cloud-based software

Enterprise web development usually presupposes building apps using cloud computing, because it doesn't restrict you to a certain location. It means that no matter where you or your team members are or what kind of device they use, they can easily access your corporate app.

Beyond that, a cloud-based management system allows for organizational scalability, so you can hire developers from distant locations and manage them just like your on-premises employees.

– Automation of administration processes

With enterprise website solutions, corporations forget about manual entry and sorting of data. The system does this for them. That's why your managers will no longer have to update the client information, payment details, accounting data, and other useful information.

Moreover, such business applications automatically upload and streamline your Excel sheets, so that you don't lose the previous data.

– App design adaptability

Intuitive design is one of the advantages of all enterprise apps that let your team members use the software without problems. However, these enterprise-level applications must also be adaptive in terms of adjustments to make this corporate management system integration barrier-free.

2. Optimal features

– Security of corporate data

Custom enterprise web development not only offers automated and adaptable data entry and organization but also safekeeping and exchange of this information. Protective measures of this software cover the defense of your sensitive data against malware, cyberattacks, ransomware, and other malicious activities.

– Recovery of information

Another essential feature of flawless enterprise web application performance is the ability to restore your data in case of an unexpected system shutdown or other circumstances. As a business leader, you can rest assured that your data will not be lost, damaged, or deleted forever.

– Mobile compatibility

How to develop an enterprise web application and make it convenient for the whole team? You should also create a version of this application for smartphones. This way your developers and other employees will be able to enter or obtain the needed information anytime and anywhere.

3. Top features

– Analytical tools

It's great to have analytics experts in place who report to you about the latest operational tendencies and help you make rational business decisions. Nevertheless, it's more reliable to have a smart application that does the same thing using embedded data analytics – and people who properly interpret this data and provide important insights.

– Digital payments support

To simplify payments processing and accounting, some enterprise web development services incorporate online payment methods. They enable you or your company managers to make payments at any time and track your expenses in one place. You won't even have to make any entries yourself since your app will do it for you.

– Social engagement functionality

Enterprise web application engineering can foster more social interactions among your developers or clients. This will help them be in touch and even find common ground, which creates trust. And trust improves your chances of striking a profitable deal with your customers.

7. Describe Concurrency and Concurrent organization

Concurrency refers to the ability of a system to execute multiple tasks or processes simultaneously. In a concurrent system, two or more tasks can be executed simultaneously, resulting in more efficient use of resources and improved performance.

A **concurrent organization** is a type of organization in which multiple teams or individuals work on different tasks or projects simultaneously. The goal of concurrent organization is to reduce the time needed to complete a project by overlapping different stages of the project. In a concurrent organization, tasks can be completed simultaneously, rather than in a sequential manner.

There are several advantages to concurrent organization:

1. **Faster time to market:** By overlapping different stages of a project, concurrent organization can reduce the time needed to complete a project and bring products to market faster.
2. **Improved efficiency:** Concurrent organization allows for more efficient use of resources and improves overall productivity.
3. **Better risk management:** By dividing a project into smaller tasks and executing them simultaneously, concurrent organization can reduce the impact of potential risks and improve risk management.
4. **Increased innovation:** Concurrent organization encourages collaboration and communication between teams, which can lead to increased innovation and creativity.

However, there are also some challenges associated with concurrent organization, such as:

1. **Coordination:** Coordinating multiple teams and ensuring that they are working towards the same goals can be challenging.
2. **Communication:** Effective communication is critical in a concurrent organization, as it is important to ensure that all teams are aware of each other's progress and any changes in project requirements.
3. **Resource allocation:** Proper resource allocation is crucial in a concurrent organization, as it is important to ensure that each team has the resources it needs to complete its tasks.

8. Explain organizing domain logic in detail with suitable diagram.

Domain logic (aka business logic, business rules, and domain knowledge) is the logic that makes business-critical decisions.

By organizing functionalities in a logical and structured way, it will be easier for developers to understand how the system works and make changes to it as needed.

Imagine that you are building a software system to manage a library. Some of the domain logic for this system include functions for checking out books, adding new books to the catalog, and generating reports on the library's collection...

Transaction Script

Transaction Script is a design pattern that organizes domain logic as a series of procedures, or "scripts," that are executed in a specific order to accomplish a particular task.

For example, one of the tasks that the system needs to be able to perform is checking out books to patrons, and it might organize this functionality as a series of scripts, such as:

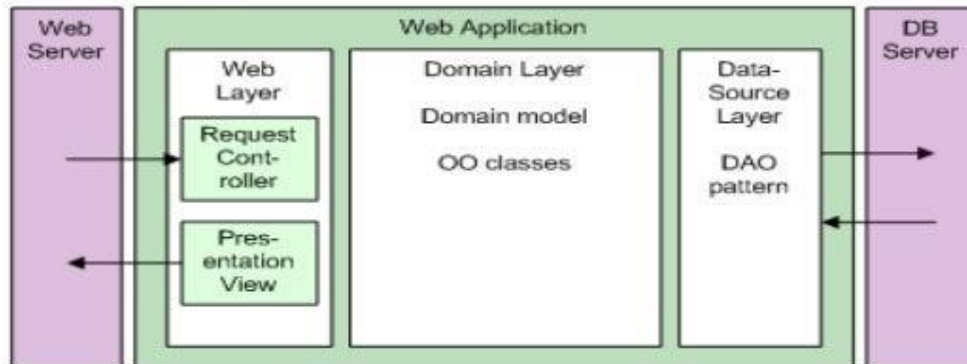
- A script to look up a patron's account based on their ID
- A script to check if the book is available to be checked out
- A script to update the book's availability status in the library's catalog

Each of these scripts is a separate procedure that performs a specific operation and is executed in a particular order to check out a book.

Transaction Script is often used in simple systems where the domain logic is relatively straightforward, and there are few relationships between different parts of the system. Still, it can become unwieldy in systems with more complex domain logic, and often there will be duplication of scripts as several transactions need to do similar things.

Domain Layer

- Provides the business logic of the application
 - Not part of the Presentation Layer nor the Data Source layer



A Domain Model is a design pattern that organizes domain logic as a set of objects representing real-world concepts and processes in the system's business domain. Each object has its behavior and state, and the objects interact with each other to accomplish tasks.

Using the Domain Model pattern, you might define objects such as "Patron," "Book," and "Loan" to represent the real-world concepts of a patron, a book, and a loan, respectively.

To check out a book, the system would create a "Loan" object and set its attributes (e.g., the patron's ID, the book's ID, and the due date). The "Loan" object would then interact with the "Patron" and "Book" objects to update their state (e.g., the "Patron" object's list of loans would be updated to include the new loan, and the "Book" object's availability status would be set to "checked out").

It allows for complex relationships between different parts of the system to be captured and represented, making the system more flexible and easier to adapt to changing requirements. However, it can be more difficult to understand and implement since it involves creating and managing multiple objects and their relationships.

Table Model

Table Module is a design pattern that organizes domain logic as a set of "table modules," which are classes that represent a specific database table and provide methods for querying and updating the data in that table.

Using the Table Module pattern, you might define a "BooksTableModule" class that represents the "books" table and provides methods such as "getBookById" and "updateAvailabilityStatus" to query and update the data in the table.

Table Module is often used in systems that use a database as their primary data store and where the domain logic is relatively simple and focused on reading and writing data to the database.

It can be easier to understand and implement, with the ability to reuse table modules across different parts of the system. However, it lacks flexibility and scalability. It can become inefficient with large volumes of data.

How do you choose between the three patterns?

Ultimately, the choice between these patterns will depend on the specific needs and constraints of the system, such as the complexity of the system's logic and the number of relationships between different parts of it, as well as the preferences and expertise of the development team.

Unit 4 Questions

1. State the key elements of Enterprise Integration

Enterprise Integration refers to the process of connecting different systems and applications within an organization in order to improve communication, collaboration, and efficiency. The key elements of Enterprise Integration are:

- **Application programming interfaces (APIs)** process data transfers between different systems. Situated between an application and web server, they enable companies to share the data and functionality of their applications with third-party developers, business partners and internal departments. With APIs increasingly used to access and expose real-time data, this can be extended to more sources, such as data published as events.
- **Application integration** is the enablement of individual applications — each designed for a specific purpose — to work collaboratively. By making it easier to share data and combine workflows and processes, organizations can benefit from integrations that modernize infrastructures without rework. Furthermore, application integration helps on-premises systems and cloud-based enterprise systems like CRMs and ERPs interact successfully without major changes to existing applications.
- **Messaging** helps provide resilience and performance to IT environments spanning cloud and on-premises systems. Messaging must cross network boundaries to provide reliable delivery while preserving network-wide message integrity, data protection and regulatory compliance via security-rich functions.
- **Events** are records of action or change. When one application or service performs an action or undergoes a change relative to the functionality of another application or service, the first one publishes an event. Other applications or services can detect the event publication. They can then process the event, perform one or more reciprocal action or simply ignore the event.
- **Data**, specifically real-world operational data, enables continuous improvement (CI) of enterprise architecture. Data is also used to assess the criticality and usage of integrations and determine their target state. When analyzed, data reveals recommended target integration patterns (e.g., service-oriented architecture (SOA), event-driven, message-driven, etc.), consolidation possibilities and other inputs that help define the target integration state.

2. Classify the different types of integration styles.

There are several different types of integration styles used in Enterprise Integration, including:

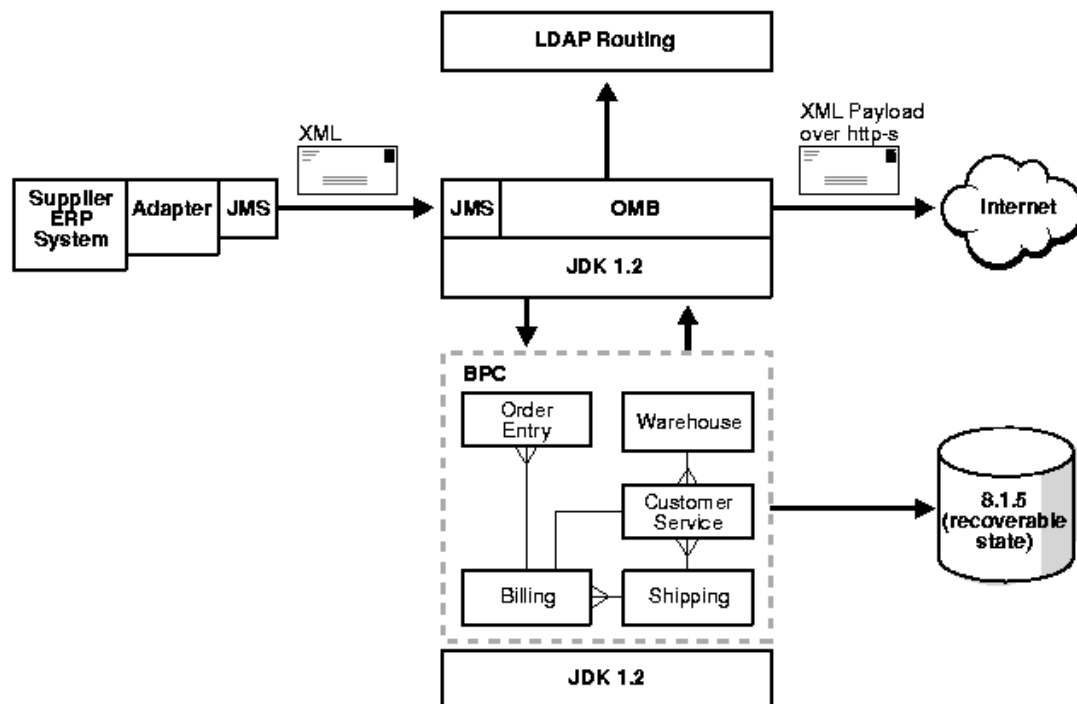
1. **Point-to-Point Integration:** This style of integration involves connecting two systems directly, without any intermediary components. It is a simple and direct approach, but it can become complex and difficult to maintain as the number of systems increases.
2. **Message-Oriented Middleware (MOM):** This style of integration involves using a message broker or middleware to facilitate communication between systems. Messages are sent between systems through the middleware, which acts as an intermediary.
3. **Service-Oriented Architecture (SOA):** This style of integration involves building systems and applications as a collection of services that can be accessed and used by other systems and applications. Services are accessed through standard interfaces, making it easier to integrate different systems.
4. **Event-Driven Architecture (EDA):** This style of integration involves using events and messages to trigger actions and processes within systems. Events can be generated by different systems and applications, and they are used to trigger actions within other systems.
5. **API-Led Integration:** This style of integration involves building systems and applications as a set of APIs that can be accessed and used by other systems and applications. APIs are accessed through standard interfaces, making it easier to integrate different systems.
6. **File-Based Integration:** This style of integration involves exchanging files between systems using a shared file system or a file transfer protocol. This approach is simple and

straightforward, but it can become complex and difficult to manage as the number of systems increases.

3. Illustrate the idea of Supplier and exchange perspective in integrating the B2B exchange

Exchange Integration Scenario: Supplier Perspective

Various steps and integration components link the supplier's supply chain application with the business-to-business exchange. In this scenario this is an Oracle exchange. The steps and components involved in establishing such connectivity are illustrated in Figure 1.

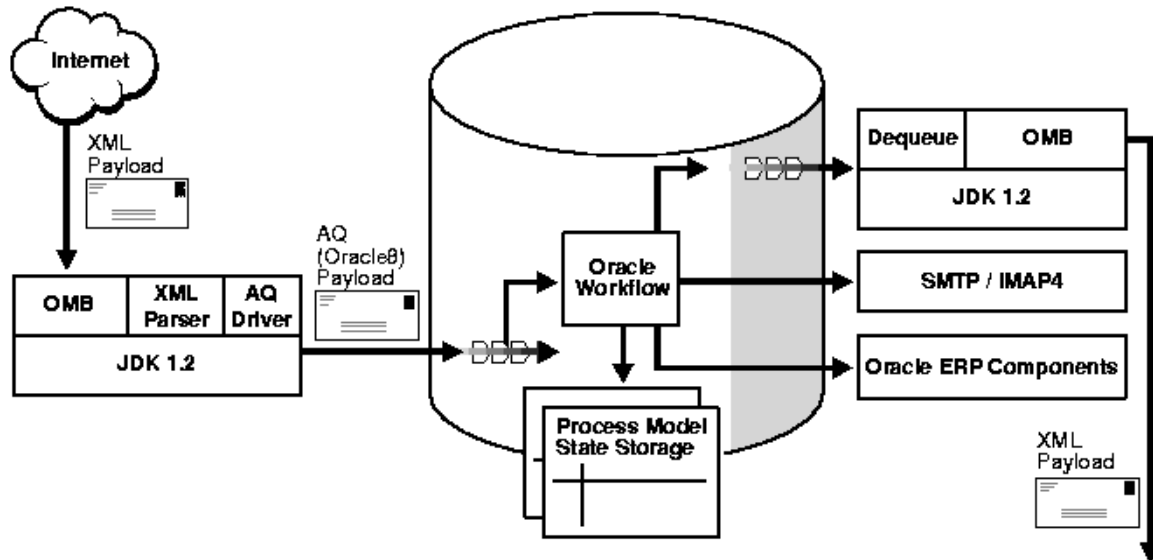


These steps and components include:

- **Adapter technology:** For the application to connect to the exchange, the supply chain application must first use an adapter to post a message to the exchange integration facility. The adapter waits for new inventory status information to be posted on a specific message queue. The adapter receives the message in the specific data format of the application, so the adapter then converts the data into XML.
- **Message propagation infrastructure:** The application adapter first enqueues the message into a message propagation infrastructure through a standard messaging interface. The messaging infrastructure can then provide three additional services:
 - It can route the message to the exchange either statically or based on its subject or content.
 - It can consult an LDAP directory service to determine where to send the message.
 - It can optionally store the message in a persistent store so that the message can be audited or tracked.
- **Local Workflow Processing:** If local processing is necessary, the local business process management facility can dequeue the message from the message propagation infrastructure, process it locally, and, when this is completed, place the message back in the message propagation infrastructure in order to send the message to the exchange.

Exchange Integration Scenario: Exchange Perspective

The integration architecture of the Oracle exchange resembles the architecture described in the previous section.



An XML payload is received over the wire on the HTTP-S protocol and goes through the following stages:

- **Message receipt and propagation:** The message management facility receives the message as an XML payload over HTTP-S.
- **Data transformation and parsing services:** When the facility receives the message, it frequently places the message in a persistent store so that it can be audited and tracked for conflict resolution. The message header and payload may often need to be parsed using an XML parser for two purposes:
 - To determine where to route information based on either the subject or topic of the message or the content of the payload. For instance, the message may need to be sent through a specific approval cycle if the supplier is posting inventory levels that are below the threshold for the exchange.
 - To parse the message into a structured format to enable it to be placed in a business process management or workflow facility for processing
- **Business Process Management:** The workflow system dequeues the message from the message propagation infrastructure and carries out an entire workflow process to update the various applications that form part of the exchange. The workflow facility can also:
 - For example, send an e-mail message to notify a small supplier by directly calling its own SMTP/IMAP4 interface
 - Send an XML message outbound: If the workflow system needs to send an XML message outbound to another supplier over HTTP-S, it can serialize the message into the XML payload format appropriate to the target, determine how to route the message by consulting an LDAP directory, and then enqueue the message into the message propagation infrastructure.

4. Describe the most common automated integration option orchestration.

Write the pros and cons of using the full automation process for integration.

The most automated integration option is orchestrations. If you are not familiar with orchestrations, they refer to the process of automating multiple systems and services together. Teams will often use software configuration management tools such as PowerShell to build orchestrations. Software configuration management tools offer various methods such as snap-ins or hosting APIs to connect with applications to manage the automation workflow.

Pros:

- Full Automation: Automation across all processes.
- Manages Multiple Systems: Ability to manage the integrations of multiple systems collectively.

Cons:

- Code-Intensive: You need to have coding skills to manage your software configuration management tool.
- Labor-Intensive: Because the workflows are quite complex, the setup can be a drawn-out process. Also, any asset or process changes force you to check how it will affect your orchestrations.

5. Explain the concept of Web Services. What are the characteristics of WSDL?

Web Services Description Language (WSDL) is a standard specification for describing networked, XML-based services. It provides a simple way for service providers to describe the basic format of requests to their systems regardless of the underlying run-time implementation.

WSDL defines an XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are first described abstractly and then bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages, regardless of which message formats or network protocols are used to communicate. This means that interfaces are defined abstractly using XML schema and then bound to concrete representations that are appropriate for the protocol.

WSDL allows a service provider to specify the following characteristics of a Web service:

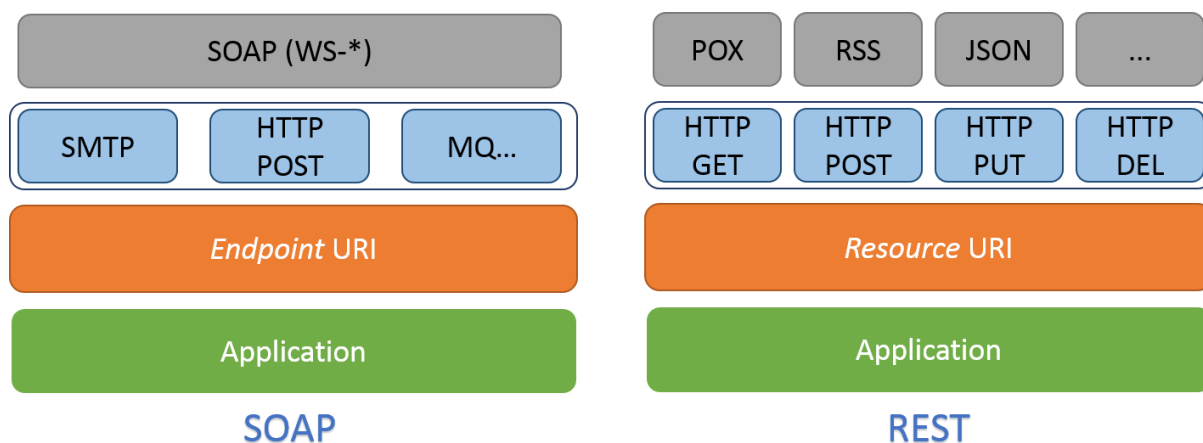
- The name of the Web service and addressing information
- The protocol and encoding style to be used when accessing the public operations of the Web service
- The type information such as operations, parameters, and data types comprising the interface of the Web service

6. Differentiate between SOAP and REST.

#	SOAP	REST
1	A XML-based message protocol	An architectural style protocol
2	Uses WSDL for communication between consumer and provider	Uses XML or JSON to send and receive data
3	Invokes services by calling RPC method	Simply calls services via URL path
4	Does not return human readable result	Result is readable which is just plain XML or JSON
5	Transfer is over HTTP. Also uses other protocols such as SMTP, FTP, etc.	Transfer is over HTTP only
6	JavaScript can call SOAP, but it is difficult to implement	Easy to call from JavaScript
7	Performance is not great compared to REST	Performance is much better compared to SOAP - less CPU intensive, leaner code etc.

SOAP	REST
1. Simple Object Access Protocol	1. Representational State Transfer
2. Function-driven (data available as services, e.g.: "getUser")	2. Data-driven (data available as resources, e.g. "user").
3. Message format supported: XML	3. Message format supported: Plain text, HTML, XML, JSON, YAML, etc.
4. Transfer Protocols supported: HTTP, SMTP, UDP, etc.	4. Transfer Protocols supported: HTTP
5. SOAP is recommended for Enterprise apps, financial services, payment gateways	5. REST is recommended for mobile applications and Social networking applications.
6. Highly secure and supports distributed environment.	6. Less secured and not suitable for distributed environment.

Fig. SOAP vs REST



7. Describe the services provided by restful API servers.

Restful API servers provide a range of services that enable clients to interact with resources over the web using the REST architectural style. Some of the key services provided by RESTful API servers include:

1. Resource identification: RESTful APIs use URIs to identify resources, which can be any type of data, including documents, images, videos, or data records.
2. Resource manipulation: RESTful APIs provide a set of HTTP methods (GET, POST, PUT, DELETE) that allow clients to create, read, update, or delete resources. These methods are mapped to CRUD (Create, Read, Update, Delete) operations in a database.
3. Stateless communication: RESTful APIs are stateless, meaning that each request contains all the information necessary for the server to process the request. This enables the server to scale easily and handle large volumes of requests.
4. Representation of resources: RESTful APIs represent resources using a variety of formats, such as JSON or XML. This allows clients to choose the format that best suits their needs.
5. Caching: RESTful APIs support caching of responses, which can improve performance and reduce network traffic.
6. Security: RESTful APIs provide a range of security mechanisms, including authentication, authorization, and encryption, to ensure that resources are accessed only by authorized clients.
7. Discoverability: RESTful APIs provide a uniform interface that enables clients to discover available resources and their capabilities.

Overall, RESTful APIs provide a flexible, scalable, and efficient way for clients to interact with resources over the web, making them an essential component of modern web applications and services.

8. Give two modern service integration techniques with their advantages and disadvantages.

Two modern service integration techniques are:

1. Event-Driven Architecture (EDA):

EDA is a service integration technique where communication is based on the events. An event is a change in state or a trigger in a system. EDA follows a publish-subscribe model, where publishers send events to subscribers, and subscribers listen to specific events that they are interested in. EDA provides benefits such as scalability, flexibility, and responsiveness. However, implementing EDA requires careful planning, and it can be challenging to manage events in large-scale systems.

Advantages:

- Scalability: EDA allows scaling of individual services without affecting other services in the system.
- Flexibility: EDA provides a flexible architecture that can handle various event sources and types.
- Responsiveness: EDA enables real-time processing of events, leading to quicker responses.

Disadvantages:

- Complexity: EDA can be complex to design and implement.
- Event Management: Managing events in a large-scale system can be challenging.
- Debugging: Debugging issues in EDA can be challenging, as events can be asynchronous and distributed.

2. GraphQL:

GraphQL is a query language and runtime for APIs that was developed by Facebook. It provides a flexible and efficient way to request and deliver data. In GraphQL, clients define the structure of the data they need, and the server delivers the exact data requested. GraphQL allows retrieving multiple resources in a single request, reducing the number of requests needed to the server. GraphQL provides benefits such as efficiency, flexibility, and improved developer experience. However, it requires a learning curve to master and can lead to over-fetching and

under-fetching of data.

Advantages:

- Efficiency: GraphQL allows retrieving multiple resources in a single request, leading to improved performance and reduced network traffic.
- Flexibility: GraphQL provides a flexible data model that allows developers to request only the data they need.
- Improved Developer Experience: GraphQL provides tools for client-side development, making it easier for developers to build and maintain applications.

Disadvantages:

- Learning Curve: GraphQL requires a learning curve to master, especially for developers who are new to the technology.

Over-fetching and Under-fetching of Data: GraphQL can lead to over-fetching and under-fetching of data, as the client defines the structure of the data they need.

Unit 5 Questions

1. Express the idea of deployment of an application? Explain with example.

Deployment of an application refers to the process of making an application available to users or clients after its development. It involves setting up the necessary infrastructure and configurations to run the application in a production environment.

For example, let's consider the deployment of a web application that allows users to book movie tickets. After the development phase is completed, the application needs to be deployed to a web server so that users can access it through their web browsers.

The deployment process typically involves the following steps:

1. Infrastructure setup: This involves setting up the necessary hardware, software, and network components required to run the application. This includes servers, databases, load balancers, firewalls, and other related tools.
2. Configuration: Once the infrastructure is set up, the next step is to configure the application to run in the production environment. This includes setting up the web server, configuring the database, and configuring any other software components that the application requires.
3. Testing: Before making the application available to users, it is important to test it thoroughly to ensure that it is working correctly. This may involve testing the application's performance, security, and functionality.
4. Deployment: Once the application has been tested and is ready for production use, it can be deployed to the web server. This may involve copying the application files to the server, configuring the web server to run the application, and starting the application.
5. Maintenance: Once the application is deployed, it needs to be maintained and monitored to ensure that it continues to run smoothly. This may involve performing regular backups, applying security patches, and monitoring the application's performance.

In summary, deploying an application involves setting up the infrastructure, configuring the software components, testing the application, and making it available to users in a production environment.

2. Compare Group Policy, PDQ Deploy and CITRIX with its pros and cons.

Feature	Group Policy	PDQ Deploy	CITRIX
Definition	A feature in Microsoft Windows that allows administrators to manage and configure settings on multiple computers in an organization	A software deployment tool that simplifies the process of installing and updating software across multiple computers	A virtualization solution that allows users to access applications and desktops from anywhere
Deployment	Centralized control	Centralized control	Decentralized control
Targeted Devices	Windows OS devices	Windows OS devices	Various OS and devices
Scalability	Limited to Windows	Limited to Windows	Scalable to multiple

	environment	environment	devices
Security	Secure communication	Secure communication	Secure virtualization
Automation	Automatic application	Automatic application	Automatic deployment
Management	Limited to Windows environment	Limited to Windows environment	Management of virtualization
Cost	No additional cost	Paid software	Paid software
Pros	Secured way of deploying over a network	Applications features can be edited before deploying	Suitable for apps and mobile applications
Cons	Organizing and handling large number of GPOs is difficult	Features like automated Windows update repository are missing	It is a costly solution

3. How security can be achieved while doing enterprise deployment?

Enterprise deployment often involves installing software on multiple machines within an organization. This process needs to be done securely to ensure that sensitive data is not compromised. Here are some ways to achieve security during enterprise deployment:

1. **Use secure communication channels:** When deploying software, it is important to ensure that the communication channel between the deployment server and the client machines is secure. This can be done by using secure protocols such as HTTPS, SSH, or SSL.
2. **Authentication and Authorization:** Authentication is the process of verifying the identity of a user, while authorization is the process of granting or denying access to a particular resource. Strong authentication and authorization mechanisms should be put in place to prevent unauthorized access.
3. **Encryption:** Data should be encrypted both in transit and at rest. This prevents unauthorized access to data by hackers or other malicious actors.
4. **Access control:** Access control policies should be put in place to ensure that only authorized personnel have access to the deployment environment. This can be done by implementing role-based access control or other access control mechanisms.
5. **Testing and Auditing:** It is important to thoroughly test the deployment process and audit the deployed software for vulnerabilities. Regular security testing should be conducted to identify and fix security flaws.
6. **Patching and Updates:** Regular patching and updating of software is essential to ensure that vulnerabilities are addressed promptly.

By implementing these security measures, an organization can ensure that their enterprise deployment is secure and their sensitive data is protected.

4. Explain the best-known metric of network availability. Discuss the parallel transport availability.

The best-known metric of network availability is known as “five nines”. What five nines means is that the end-user perceives that their application is available 99.999% of the time. This permits only 5.26 minutes of downtime a year. Depending on the application and network topology, this can be a very stringent standard.

Parallel transport availability is a metric that measures the ability of a network to maintain continuous communication between two or more points, even in the event of a failure in one or more network links. This is achieved by creating parallel paths between the points, such that if one path fails, the data can be rerouted through another path, thus ensuring uninterrupted communication.

Parallel transport availability is achieved through the use of redundancy and failover mechanisms in the network infrastructure. Redundancy refers to the provision of multiple paths between the points, while failover refers to the ability to automatically switch to an alternate path in the event of a failure in the primary path.

The parallel transport availability metric is usually expressed as a percentage, which represents the amount of time that the network is available for parallel transport between the points. For example, a network with a parallel transport availability of 99.999% (five nines) would be available for parallel transport for 99.999% of the time, or 5 minutes and 15.6 seconds of downtime per year.

Parallel transport availability is critical for enterprise systems that require high availability, such as financial transactions, healthcare systems, and emergency services. By ensuring continuous communication between the points, parallel transport availability helps to prevent data loss, reduce downtime, and minimize the impact of failures on the system.



5. Give any key requirement to successful delivery. How it can transform the work environment?

Transparency is a key requirement for successful delivery in any organization. It involves clear communication of information, decisions, and actions to all stakeholders involved in the project.

Transparency can transform the work environment in several ways:

6. **Trust:** Transparency builds trust between team members and stakeholders. When everyone has access to the same information, there are no hidden agendas or surprises. This can help to build trust and foster better working relationships.
7. **Collaboration:** Transparency encourages collaboration between team members. When everyone is aware of what is going on, they can provide feedback, suggest improvements, and work together to solve problems.
8. **Accountability:** Transparency helps to hold team members accountable for their actions. When everyone can see what is going on, it is easier to identify who is responsible for specific tasks or decisions.
9. **Continuous Improvement:** Transparency allows for continuous improvement. When everyone has access to the same information, it is easier to identify areas for improvement and make changes to the process.

Unit 6 Questions

1. Illustrate why are Enterprises moving towards cloud? Give any three reasons.

1. Cost savings

A typical enterprise cloud solution leverages pay-as-you-go pricing, so businesses only pay for the resources they use. In addition, businesses that move to the Cloud can avoid many or all of the up-front costs of developing similar capabilities in-house. There's no need to lease a data center, no servers to buy, and no physical computing infrastructure that needs to be maintained. As a result, IT expenses for enterprise cloud adopters are often lower, easier to calculate, and easier to predict.

2. Security

Enterprise organizations are frequently targeted by cyber criminals wishing to steal or expose data. Data breaches are extremely costly to remedy and can negatively impact your reputation and customer relationships. With enterprise cloud, organizations can access security tools like system-wide identity/access management and cloud security monitoring. They can easily implement network-wide identity and access controls. Cloud service providers also play a role in supporting data security in public and private deployments.

3. Disaster Recovery/Business Resiliency

Without a solid Disaster Recovery solution, business resiliency is at risk in the face of a service outage, natural disaster, or cyberattack. Lost revenue, degradation of customer trust, and even bankruptcy are possible outcomes.

4. Flexibility & Innovation

Enterprise cloud computing offers businesses the flexibility to dynamically scale their resource consumption up or down as needed. This minimizes the amount of upfront capital cost associated with launching a new product or testing a new service and removes barriers to innovation.

2. Classify the types of Private Cloud. Give examples of private cloud providers.

Depending on who manages the private cloud environment and where the cloud solution is hosted, a private cloud can be classified into four major types – virtual private cloud, managed private cloud, hosted private cloud, and on-premise private cloud. Let's look at each one in detail.

1. Virtual private cloud

A virtual private cloud (VPC) is a type of cloud model that offers the benefits of a private cloud (more control and an isolated environment) with the help of public cloud resources. Opens a new window . Although private cloud and virtual private cloud are often used interchangeably, there are many points of difference between the two.

In a traditional private cloud, a company's internal IT department acts as the service provider, and the individual business units act as tenants. In a virtual private cloud model, a public cloud provider acts as the service provider, and the cloud users act as its tenants. Simply put, a virtual private cloud is a hybrid model of cloud computing in which a private cloud solution is provided within a public cloud provider's infrastructure.

2. Managed private cloud

A managed private cloud is a type of private cloud model in which the infrastructure is not shared. It is also referred to as a dedicated or single-tenant cloud. This type of private cloud is managed by a third-party vendor. The vendor provides support, maintenance, upgrades, and even remote management of the private cloud. In some cases, vendors also manage the software applications in cloud.

3. Hosted private cloud

Hosted private cloud vendors offer cloud servers in their own data centers and are also responsible for security management. In a hosted private cloud model, users get access to additional resources, a support team, high-demand scalability options, as well as a user-friendly dashboard to assist in server management.

4. On-Premise private cloud

Unlike hosted private clouds, on-premise cloud solutions allow users to host a cloud environment internally. For such a cloud model, it is necessary to have an internal data center to host the cloud server. This type of private cloud model is very secure as they are internally hosted and managed by an organization's internal IT department. The organization, therefore, has complete control over the security, configurations, and scalability of its servers.

Examples of Private Cloud Providers

Let's glance at the services provided by some leading private cloud vendors.

1. HPE

Hewlett Packard Enterprise (HPE) has been a leader in the private cloud computing market for many years. The private cloud service provider offers customizable private cloud software and infrastructure. The HPE private cloud can be used along with a public cloud to provide a faster connection with the same security protections as a private cloud. HPE's private cloud offerings include services, hardware, and software. Its private cloud solutions include the Helion CloudSystem hardware, Helion Cloud Suite software, Helion Managed Private Cloud, and Managed Virtual Private Cloud services, among others.

2. VMware

VMware offers two types of private cloud solutions. While one solution is completely private, the other is a hybrid solution that offers an integrated stack as well as automated lifecycle management. Although VMware is best known for its virtualization software that runs many private cloud environments, it also offers a variety of other services. VMware's vRealize Suite Cloud Management Platform offers private as well as hybrid cloud management. The VMware Cloud Foundation, on the other hand, is a data center platform for private clouds.

3. Dell

Dell EMC offers two private cloud products. While one is meant for Microsoft Azure Stack, the other is a turnkey developer platform. Dell, who was always a leader in the private cloud market, became an even stronger player after its merger with EMC. The company's cloud offerings include cloud management and cloud security software, virtual private cloud services, and various cloud consulting services.

4. Oracle

Oracle's Private Cloud Appliance is a scalable data center that is capable of processing mixed workloads. The company's private cloud solutions include its cloud platform, infrastructure, applications, lifecycle management tools, as well as integration services, along with managed cloud services.

5. IBM

The design of the IBM private cloud is based on open-source frameworks, including Kubernetes and Cloud Foundry. The company's private cloud solutions include IBM Systems and IBM Storage, IBM Cloud Managed Services, Cloud Manager, and Cloud Orchestrator. Apart from the private cloud vendors mentioned above, Microsoft, Cisco, NetApp, Red Hat, and AWS are also major players in the private cloud race.

3. Public Cloud is an IT model where on-demand computing services and infrastructure are managed by a third-party provider". Justify the statement with relevant example.

Public cloud refers to a type of cloud computing in which resources, such as virtual machines, storage, and applications, are made available to the general public over the internet by a third-party provider, on a pay-per-use basis.

An example of a public cloud provider is Amazon Web Services (AWS). AWS offers a range of services such as Elastic Compute Cloud (EC2) for virtual machine instances, Simple Storage Service (S3) for object storage, and Lambda for serverless computing. Any individual or organization with an internet connection can access and utilize these services on a pay-per-use basis, without having to invest in expensive hardware or software infrastructure.

Another example of a public cloud provider is Microsoft Azure. Azure offers similar services to AWS and is widely used by businesses and individuals around the world.

Public cloud services offer several benefits, such as scalability, flexibility, and cost-effectiveness, making it an attractive option for businesses and individuals looking to leverage the benefits of cloud computing without investing in expensive infrastructure.

4. Identify how public clouds can save money? Elaborate the different ways.

Using the public cloud can save businesses money in a couple of different ways:

1. **Lower equipment purchases costs:** Because employees can access and pay for cloud-based resources only when they need them, using public cloud-based desktops and applications is often less expensive than purchasing physical IT equipment or software packages that may or may not be used and will need to be maintained.
2. **Lower equipment maintenance costs:** With public cloud-based services, the cost of maintaining IT equipment is also passed on to the cloud service provider.
3. **Pay-as-you-go pricing model:** Public cloud providers typically use a pay-as-you-go pricing model, where customers only pay for the services and resources they use, rather than investing in expensive hardware upfront. This can help organizations reduce their capital expenditure (CapEx) and convert it into operational expenditure (OpEx).
4. **Economies of scale:** Public cloud providers benefit from economies of scale, as they can leverage their large infrastructure and resources to offer services at a lower cost. This can help organizations save money compared to building and maintaining their own infrastructure.
5. **Reduced maintenance and operational costs:** Public cloud providers handle the maintenance and operational costs associated with running a data center, which can be expensive for

organizations. This can include costs related to power, cooling, physical security, and staffing.

6. **Increased flexibility and agility:** Public clouds offer a high degree of flexibility and agility, allowing organizations to quickly scale up or down their resources as needed. This can help organizations save money by avoiding the need to invest in excess capacity to handle occasional spikes in demand.

It is important to note that while public clouds can save money, organizations need to be careful in selecting the right services and resources that fit their needs to avoid unnecessary costs.

5. Differentiate between hybrid cloud and multi cloud.

	Hybrid-Cloud	Multi-Cloud
Definition	A cloud computing environment that combines both public and private clouds.	A cloud computing strategy that involves using multiple cloud services from different providers.
Components	Typically consists of at least one private cloud and one public cloud.	Utilizes multiple cloud services, which can be either public or private.
Management	Managed and controlled by a single organization.	Managed and controlled by multiple organizations.
Deployment	Requires a certain level of integration and coordination between the public and private cloud components.	Deploying different services or applications to different clouds based on their requirements.
Flexibility	Provides a level of flexibility to an organization, allowing it to take advantage of the strengths of both public and private clouds.	Offers greater flexibility, as it allows organizations to mix and match services from different providers to meet their specific needs.
Cost	Can be more cost-effective than a purely public cloud approach, but may require more upfront investment in private cloud infrastructure.	Can be more expensive than a hybrid cloud approach, as it may require additional management and integration costs.
Security and Control	Provides greater control and security over sensitive data and applications by keeping them within a private cloud environment.	Can offer greater security by spreading workloads across multiple providers, reducing the risk of a single point of failure. However, it may also present challenges in terms of maintaining consistent security across multiple clouds.
Complexity	Can be more complex to manage due to the need for integration and coordination between multiple clouds.	Can also be complex, as organizations need to manage and integrate services from multiple providers.

6. Identify the types of scaling in cloud computing.

There are two main types of Cloud scalability, horizontal and vertical. The choice between these two approaches should depend on current needs and future requirements for the product and the organisation. Therefore, it is important to understand what each type has to offer.

1. **Horizontal scaling** – scaling out or in, involves adding or removing extra servers to the cloud infrastructure. Splitting traffic between two or more instances, can spread the load across more machines and therefore enhance the availability of our service. Horizontal scaling is easier to manage automatically, and easier to accomplish without downtime. Thanks to additional instances, this solution can also ensure better functioning in cases of natural disasters or major technical failures.
2. **Vertical scaling** – scaling up or down, refers to adding or diminishing power in an already existing instance. It focuses on improving memory, storage or processing power to cope with increased workloads. This approach does not require any modification of the code. However, it

may affect product performance or caused downtime. Vertical scaling allows for better optimisation of resources relative to the actual time of use, which if done correctly can help lower cloud costs.

Those scaling types do not exclude each other, and if needed they can be combined. For example, organisations can scale up vertically until the server limit is reached, and then clone the server to add further resources if necessary. This variation can be a good option for businesses with more unpredictable environments because scaling both up and down as well as in and out allows you to remain more agile.

7. Explain the concept of cloud-based software testing.

Cloud-based testing means performing tests for a software application through resources found in the cloud. These tests can include the hardware, software and infrastructure of an application. QA teams rely on a cloud software testing strategy and these cloud based mobile testing solutions in order to verify a product's security, functionality and usability before market launch.

Cloud testing concentrates on these core testing components to ensure full testing coverage:

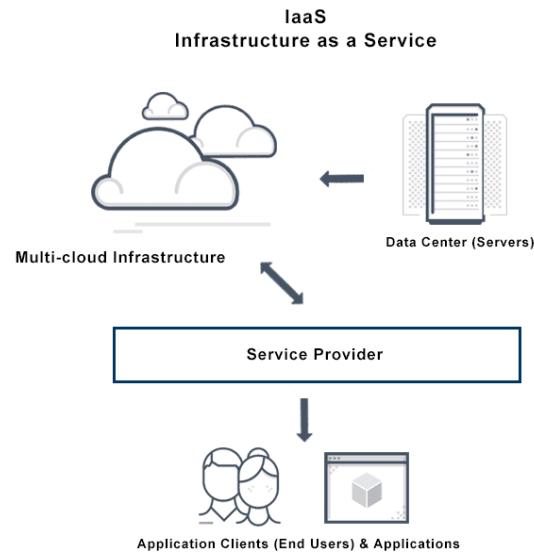
1. **Application:** Covers cloud based testing software for functionality, data security, browser compatibility and end-to-end business workflows.
2. **Network:** Includes testing a variety of network bandwidths and protocols as well as successful transfer of data through networks.
3. **Infrastructure:** Focuses on testing for disaster recovery, secure connectivity, backups, and storage policies.

How the Cloud Changes Testing

Cloud software testing is essential after migrating to the cloud. However, the cloud environment encourages QA teams to alter how they execute test cases so that they experience successful testing cycles.

1. **Functional Testing:** Cloud testing means validating the cloud service or SaaS functions, including the end-to-end functionality of an application.
2. **Integration Testing:** QA testers review SaaS based integration in the cloud as well as the application's integration between legacy systems.
3. **Security Testing:** Cloud-based mobile app testing solutions allow QA testers to execute test cases that focus on user privacy and security across a diverse range of the user. It also focuses on data integrity and protection during transit and rest periods, connectivity security, protection against cyberattacks, and the security of the software interface.
4. **Performance Testing:** Cloud based mobile app testing tools enable QA teams to leverage cloud for load testing, monitor application stability and execute performance testing in a scalable environment.

8. Explain why is IaaS important? Discuss the use cases of Infrastructure as a Service



Why is Infrastructure as a Service important?

You can use IaaS to scale your compute capacity while reducing your IT expenditure. Traditionally, enterprises purchased and maintained their own computing devices in an on-premises data center. However, this often required a heavy up-front investment to handle only occasionally high workloads. For example, an e-commerce company gets three times more application traffic during the holiday season. To handle this traffic, they have to purchase additional server machines, which remain idle for the rest of the year.

To overcome this challenge, cloud providers like AWS maintain highly secure data centers with a large volume of hardware devices. They give you access to this cloud computing infrastructure on a pay-as-you-go basis. You get flexible and secure access to practically unlimited resources so that you can meet all your business, legal, and compliance requirements.

What are the use cases of Infrastructure as a Service?

You can use cloud infrastructure to improve operational efficiency and prioritize solution delivery over infrastructure management. An IaaS provider can support you to improve customer experience with high-performing, fully managed infrastructure. Let's look at some example use cases below.

1. High performance computing

Complex problems like analyzing large volumes of data or solving physics and chemistry equations require significant computational power. It is more efficient and cost-effective to solve these problems on IaaS infrastructure instead of running your own resources.

2. Website hosting

Organizations use cloud infrastructure to host high performing web applications that are secure, scalable, and fully customizable to meet their content delivery needs. For example, Amazon Web Services (AWS) offers low-cost web hosting solutions that you can use to build a range of websites, from simple information sites to complex data delivery systems.

3. Big data analytics

Companies analyze data to derive business intelligence and actionable insights. Cloud infrastructure includes data warehousing technology to store large volumes of data in an integrated way. An IaaS provider supports big data analytics by providing cloud computing services that you can use to manage data more efficiently.

4. App development

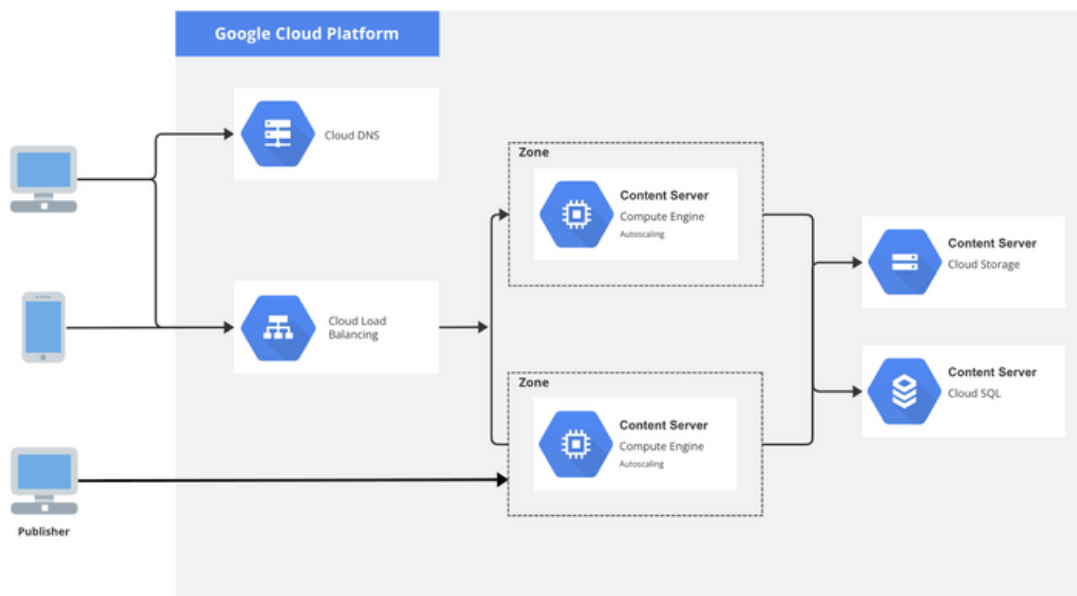
You can use cloud infrastructure to quickly set up separate test and development environments. You can experiment and test new ideas in isolation or create common development environments for the whole team.

9 List the major public clouds. Also identify the pros and cons for any two public clouds

Google Cloud Platform (GCP),

offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, Google Drive, and YouTube. Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning. Registration requires a credit card or bank account details.

Google Cloud Platform provides infrastructure as a service, platform as a service, and serverless computing environments



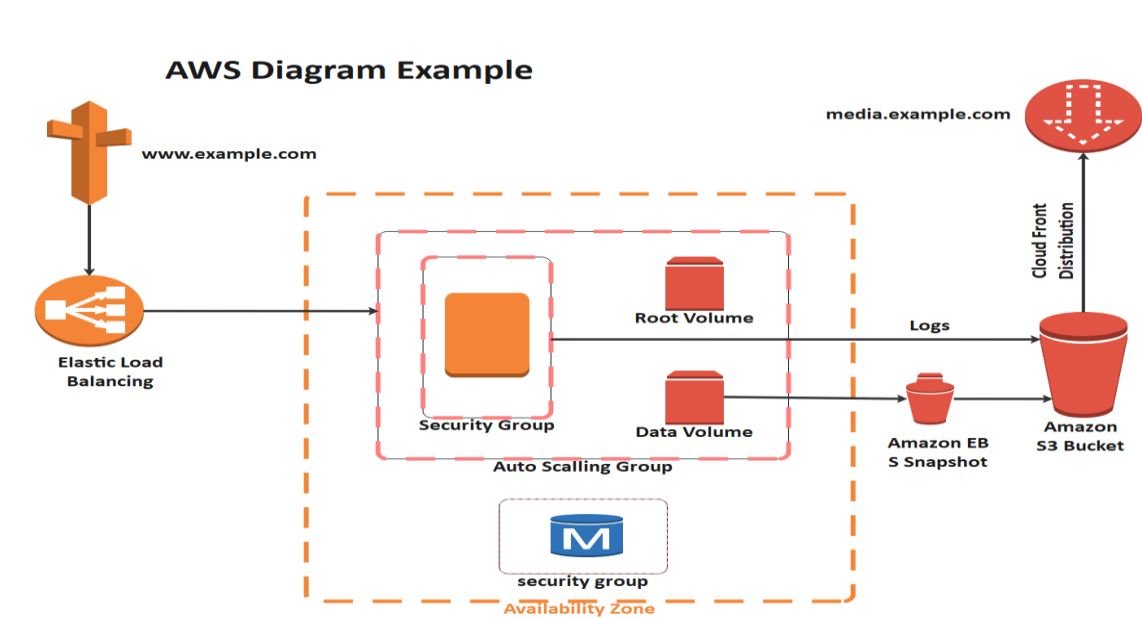
Google Cloud: Pros and cons

Strengths	Weaknesses
Excellent integration with other Google services	Majority of components based on Google proprietary tech; no real control over Virtual Machines
Fast I/O	Limited choice of programming languages
Strong data analytics and storage	Complex transition away from the platform to another vendor
Facilitates easy collaboration	Fewer features/services
Designed for cloud-native business	Fewer global data centers
Good portability and open source integration	

Amazon Web Services (AWS)

It is a platform that offers flexible, reliable, scalable, easy-to-use and, cost-effective cloud computing solutions.

AWS is a comprehensive, easy to use computing platform offered Amazon. The platform is developed with a combination of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

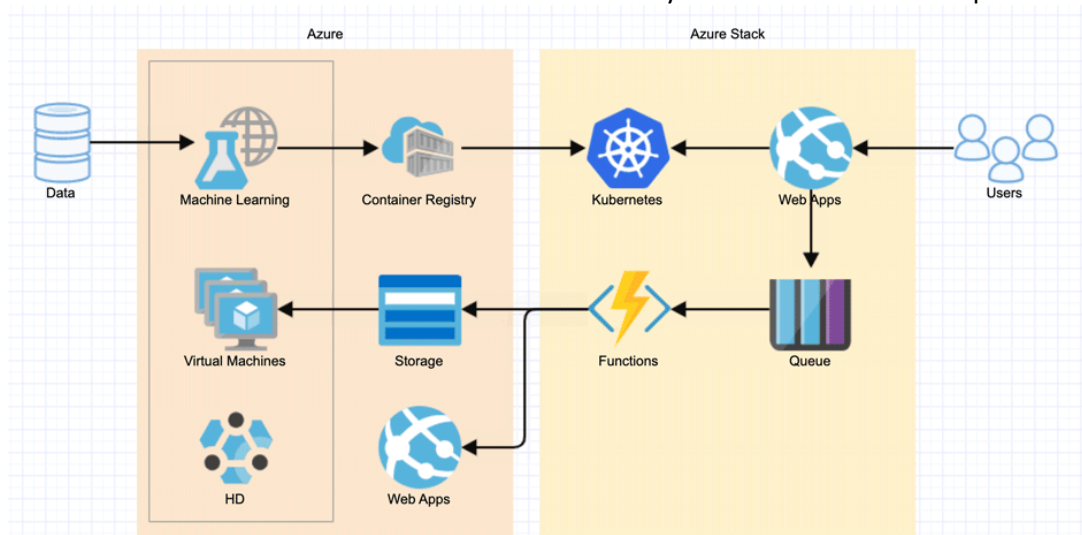


AWS: Pros and cons

Strengths	Weaknesses
Extensive range of infrastructure applications	Range of infrastructure options can be overwhelming for more traditional enterprises
Highly flexible	Hybrid options available, but not a priority
Easy transition for users with existing digital infrastructure	Organizations operating on legacy systems may experience longer migration times
Frequently updated and maintained	
Free tier available	
Greater control over security	
Scalability	
Cost-effective pricing model	
Rapid deployment	
Support for large enterprises	

Microsoft Azure

Azure is a cloud computing platform and an online portal that allows you to access and manage cloud services and resources provided by Microsoft. These services and resources include storing your data and transforming it, depending on your requirements. To get access to these resources and services, all you need to have is an active internet connection and the ability to connect to the Azure portal



Microsoft Azure: Pros and cons

Strengths	Weaknesses
High availability	Requires considerable management
Strong focus on Security	Requires platform expertise
Scalability	More limited backward compatibility
Cost-effective	Comparatively more costly than other leading vendors
Strong IaaS and PaaS options	Additional charge for pay-as-you-go option
Support for open source	Customer service
Hybrid cloud	

Unit 7 Questions

1. Describe the Enterprise architecture frameworks? Explain with the help of an example

Enterprise Architecture Frameworks (EAFs) are a set of guidelines, principles, and best practices that help organizations to create a comprehensive blueprint for managing and aligning IT resources with business objectives. They provide a structured approach for designing, implementing, and managing enterprise architecture.

Enterprise architectures are typically implemented as frameworks. There are many different frameworks, and some will be a better fit than others when it comes to any one organization. For example, a framework focused on consistency and relationships between various parts of an overarching enterprise will be more helpful to larger organizations with many moving parts compared to small ones. In this case, a framework like the Unified Architecture Framework (UAF) may work.

Some example frameworks include:

- **The Zachman Framework for Enterprise Architecture** -- which covers six architectural points as well as six primary stakeholders that aid in defining and standardizing IT architecture components.
- **Unified Architecture Framework (UAF)** -- which is a complex but flexible enterprise architecture framework suitable for military and government software development as well as use in commercial businesses. It's implemented as a UML profile.
- **Agile enterprise architecture** -- which focuses an organization around a flexible, extended collection of structures and processes that can grow. It can become an important part of Agile software delivery.
- **Federal Enterprise Architecture Framework (FEAF)** -- which is a reference model that was introduced in 1996 for IT effectiveness. It was designed for the U.S. government but can be used in private companies as well.
- Other frameworks include **The Open Group Architectural Framework (TOGAF)**, the European Space Agency Architectural Framework, the SAP Enterprise Architecture Framework or the Ministry of Defence Architecture Framework.

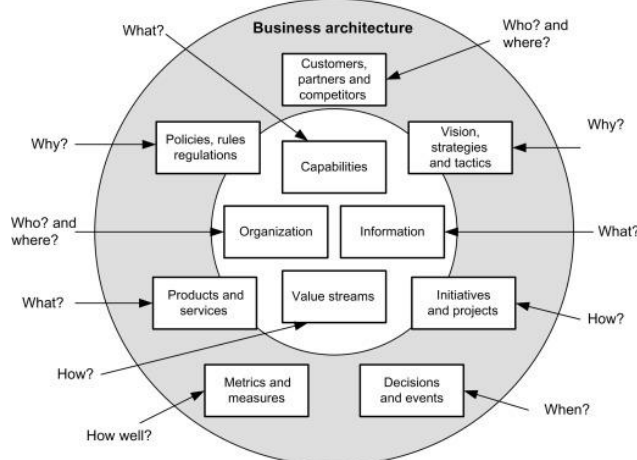
2. "Enterprise architecture facilitates collaboration with project management." Justify the statement with proper explanation.

- It provides a clear model of the organization's business, application, data and technology architecture, dependencies, and inter-relatedness. This will help the organization to make business decisions based on a holistic view instead of as a stand-alone part.
- Enterprises can increase their business values by aligning IT with their business strategy; it helps the organization to unlock the power of information, unifying information silos that inhibit business processes.
- EA ensures organizations invest in projects that are targeted towards their goals, objectives, and visions. It identifies opportunities for reuse and integration which prevents inconsistent processes and information.
- It provides an organization with a planning process to better understand its business strategy; which helps the organization to respond faster to competitive pressures and deploy a higher quality faster.
- EA identifies duplicate and overlapping processes, services, data hardware, and software, traces high-cost areas of IT assets to develop a fairer cost model, and ensures compliance with legal and regulatory laws.

3. Outline the working of enterprise architecture models with diagram.

Definition:

A compendium of the highest level of data and process models is an enterprise architecture model. This is a model that captures high-level business entities (BECs) and high-level business processes (BASs) that reflect the major reasons for the enterprise's (corporation's) existence.



How Does an Enterprise Architecture Model Work?

In the end, the frameworks outlined in an Enterprise Architecture model help the business connect with customers and partners by enabling all interactions and transactions. These interactions are being driven by the systems that make up an enterprise's architectural components. People might see them as decision-making tools and services, information assets, communication routes, and workflows. In order for individuals to connect with the company, they assume solid shape in both real and virtual settings, including private discussions, phone calls, or web-based transactions.

Enterprise Architecture projects need to be conscious of this function and create architectures that provide actual benefit to people in accessible systems rather than being viewed as a purely background operation. By creating processes that are tailored to the demands of consumers, workers, and other stakeholders, they should implement the strategic goal in a methodical way. They should provide tangible designs that people can see and use.

Applying the architecture perspective makes it easier to understand how an enterprise structure affects a design and how it incorporates architectural decisions and principles to be a part of that structure. Every design includes architecture as a fundamental element that shapes the company as a whole and greatly affects how it functions and the experiences people have with it.

4. Illustrate how can you make good enterprise architectural model? Take the use case to better illustrate the analogy.

A good Enterprise Architecture model helps you understand how your organization works, and the source of a deficiency you would like to correct. A great Enterprise Architecture model lets you exercise change and see the impact of change.

Let's consider an analogy to better illustrate our point. A budget spreadsheet will break your spending down to rent, power, heat, investments, etc. Then, if you wanted to increase spending on other things, such as a house, you can see what options you have in your spreadsheet. Enterprise architects do the same thing to look at things like changing digital customer engagement, integrating an acquisition, or upgrading major enterprise software. Really poor models are often static, as they simply "show" something. Bad models don't let you see impact of possibility, while good models do. Really strong models let you focus the purpose of change on stakeholder for decision-maker concerns like agility or other standard EA use cases.

5. Discuss the approach used by TOGAF to EAF's. List the benefits of using TOGAF.

The TOGAF approach to EAFs

TOGAF (The Open Group Architecture Framework) is an industry standard framework for enterprise architecture. It provides a common language, methodology, and tools for designing and managing enterprise architecture. TOGAF uses an iterative approach to develop the architecture, starting with a high-level view and then gradually refining it.

The Open Group defines the TOGAF as the "de factor global standard for enterprise architecture". The framework is intended to help enterprises organize and address all critical business needs through four goals:

- **Ensuring all users, from key stakeholders to team members, speak the same language.** This helps everyone understand the framework, content, and goals in the same way and gets the entire enterprise on the same page, breaking down any communication barriers.
- **Avoiding being "locked in"** to proprietary solutions for enterprise architecture. As long as the company is using the TOGAF internally and not towards commercial purposes, the framework is free.
- **Saving time and money** and utilizing resources more effectively.
- **Achieving demonstrable return on investment (ROI).**

Benefits of using TOGAF

The benefits of Architecture Development Method(ADM) are that it is customizable to organizational need—there's no need to create a structure that doesn't serve your business. These smaller packages are also scalable, so if one team rolls it out, it can successfully be rolled out to other teams without much tweaking. This helps the enterprise establish a process with multiple check points, so that there are few errors the wider the architecture is implemented.

There can also be benefits to individuals who certify in TOGAF. A study of industry employees indicates that enterprise architects, software architects, and IT directors, among others, who choose to earn a certification in TOGAF often see an average yearly pay bump of \$10,000 to \$20,000 over similarly placed colleagues who aren't certified.

Some experts in enterprise architecture point out that while TOGAF may appear very logical, it's actually quite a shake up to traditionally educated technology consultants today – but perhaps this will change as TOGAF adoption continues along steadily.

6. Describe in briefly:

a. Zachman Framework

What is Zachman Framework?

Enterprise Architecture (EA) is a discipline which has evolved to structure the business and its alignment with the IT systems. The Zachman Framework is an enterprise ontology and is a fundamental structure for Enterprise Architecture which provides a way of viewing an enterprise and its information systems from different perspectives, and showing how the components of the enterprise are related.

Why Zachman Framework?

In today's complex business environments, many large organizations have great difficulty responding to changes. Part of this difficulty is due to a lack of internal understanding of the complex structure and components in different areas of the organization, where legacy information about the business is locked away in the minds of specific employees or business units, without being made explicit.





































The Zachman framework provides a means of classifying an organization's architecture. **It is a proactive business tool, which can be used to model an organization's existing functions, elements and processes - and help manage business change.** The framework draws on Zachman's experience of how change is managed in complex products such as airplanes and buildings.

Structure of Zachman Framework

Zachman Framework is a two-dimensional classification scheme for descriptive representations of an Enterprise that is structured as a matrix containing **36 cells**, each of them focusing on one dimension or perspective of the enterprise. Rows are often presented as different viewpoints involved in the systems development process, while columns represent different perspectives of the stakeholders involved in the organization.

The rows of Zachman Framework focus on describing the enterprise from six viewpoint perspectives of the stakeholders. These six perspectives are based on English language interrogatives 'what', 'where', 'who', 'when', 'why', and 'how' (known as W5H).

The columns of the framework consist of a set of artifacts which are description of the enterprise from specific viewpoint of a group of stakeholders. The stakeholders are generally grouped as planners, owners, designers (architects), implementers, sub-constructors, users, or sometimes represented as viewpoints: scope context, business concepts, system logic, technology physics, component assemblies and operations classes.

	WHAT	HOW	WHERE	WHO	WHEN	WHY	
SCOPE CONTEXTS	Inventory Identification  Inventory Types	Process Identification  Process Types	Network Identification  Network Types	Organization Identification  Organization Types	Timing Identification  Timing Types	Motivation Identification  Motivation Types	STRATEGISTS AS THEORISTS
BUSINESS CONCEPTS	Inventory Definition  Business Entity Business Relationship	Process Definition  Business Transform Business Input	Network Definition  Business Location Business Connection	Organization Definition  Business Role Business Work	Timing Definition  Business Cycle Business Moment	Motivation Definition  Business End Business Means	EXECUTIVE LEADERS AS OWNERS
SYSTEM LOGIC	Inventory Representation  System Entity System Relationship	Process Representation  System Transform System Input	Network Representation  System Location System Connection	Organization Representation  System Role System Work	Timing Representation  System Cycle System Moment	Motivation Representation  System End System Means	ARCHITECTS AS DESIGNERS
TECHNOLOGY PHYSICS	Inventory Specification  Technology Entity Technology Relationship	Process Specification  Technology Transform Technology Input	Network Specification  Technology Location Technology Connection	Organization Specification  Technology Role Technology Work	Timing Specification  Technology Cycle Technology Moment	Motivation Specification  Technology End Technology Means	ENGINEERS AS BUILDERS
COMPONENT ASSEMBLIES	Inventory Configuration  Component Entity Component Relationship	Process Configuration  Component Transform Component Input	Network Configuration  Component Location Component Connection	Organization Configuration  Component Role Component Work	Timing Configuration  Component Cycle Component Moment	Motivation Configuration  Component End Component Means	TECHNICIANS AS IMPLEMENTERS
OPERATIONS CLASSES	Inventory Instantiation  Operations Entity Operations Relationship	Process Instantiation  Operations Transform Operations Input	Network Instantiation  Operations Location Operations Connection	Organization Instantiation  Operations Role Operations Work	Timing Instantiation  Operations Cycle Operations Moment	Motivation Instantiation  Operations End Operations Means	WORKERS AS PARTICIPANTS
	INVENTORY SETS	PROCESS TRANSFORMATIONS	NETWORK NODES	ORGANIZATION GROUPS	TIMING PERIODS	MOTIVATION REASONS	

The framework enables complex subjects to be distilled into systematic categories in the column headers, using these six basic questions (known as 5WH). The answers to these questions will differ, depending on the perspective or audience (represented in the rows).

Each view is a description from a particular perspective and has a representation (a model or functioning system), as indicated in the Table above. Here is a brief description of each view and model/functioning system:

Columns of Zachman Framework

The columns represent the interrogatives or questions that are asked of the enterprise. These are:

- **What** (data) - what is the business data, information or objects?
- **How** (function) - how does the business work, i.e., what are the business' processes?
- **Where** (network) - where are the businesses operations?
- **Who** (people) - who are the people that run the business, what are the business units and their hierarchy?
- **When** (time) - when are the business processes performed, i.e., what are the business schedules and workflows?
- **Why** (motivation) - why is the solution the one chosen? How was that derived from? What motivates the performance of certain activities?

Rows of Zachman Framework

Each row represents a distinct view of the organisation, from the perspective of different stakeholders. These are ordered in a desired priority sequence. A row is allocated to each of the following stakeholders:

- **Planner's View** (Scope Contexts) - This view describes the business purpose and strategy, which defines the playing field for the other views. It serves as the context within which the other views will be derived and managed.
- **Owner's View** (Business Concepts) - This is a description of the organization within which the information system must function. Analyzing this view reveals which parts of the enterprise can be automated.
- **Designer's View** (System Logic) - This view outlines how the system will satisfy the organization's information needs. The representation is free from solution specific aspects or production specific constraints.
- **Implementer's View** (Technology Physics) - This is a representation of how the system will be implemented. It makes specific solutions and technologies apparent and addresses production constraints.
- **Sub-Constructor's View** (Component Assemblies) - These representations illustrate the implementation-specific details of certain system elements: parts that need further clarification before production can begin. This view is less architecturally significant than the others because it is more concerned with a part of the system than with the whole.
- **User's View** (Operations Classes) - This is a view of the functioning system in its operational environment.

Rules of Zachman Framework

The framework offers a set of descriptive representations or models relevant for describing an enterprise.

- Each cell in the Zachman Framework must be aligned with the cells immediately above and below it.
- All the cells in each row also must be aligned with each other.
- Each cell is unique.
- Combining the cells in one row forms a complete description of the enterprise from that view.

b. Importance of enterprise architecture

Enterprise architecture will help multiple departments in a business understand the broader business model and articulate challenges and business risks. Because of this, enterprise architecture has an important role in unifying and coordinating departmental processes across an organization. Being able to access and understand business capability should also help individuals identify gaps in their business, and from there, they can make more informed decisions.

The definitions of EA emphasize EA as a framework and EA as a process for transforming an enterprise. The increasing pace of information technology has influenced the increased need for Enterprise Architecture. Adopting EA is the key to the survival of an enterprise due to the high rates of change and complexity in the world economy.

An enterprise that aspires to achieve its vision must be able to identify. Its current or as-it state and have a concrete plan on how to get to its target or to-be state. Without an appropriate communication method and tools, it can be challenging to communicate the vision of the enterprise.

However, EA depicts an enterprise's current state and aspired future state with visual models making communication much easier and faster. Enterprise Architecture plays an important role in an organization. It is critical to the survival and success of the organization while enabling the organization to achieve the right balance between IT efficiency and business innovation.

Typically, EA helps to facilitate business success such as competitive advantage through the effective use of information management strategies and IT resources. Enterprise Architecture can use by a company to organize and structure. Its enterprise infrastructure providing stakeholders and system architects with appropriate architectural details.

Enterprise Architecture may, however, develop for a wide variety of reasons. EA develops for:

Alignment:

To ensure that the implemented enterprise aligns with management's intent.

Integration:

The connectivity and interoperability of business rules, processes, information flow, and interfaces are consistent across the organization.

Convergence:

Pushing towards a standardized IT portfolio based on the Technical Reference Model (TRM). Thus, creating a common organizational language.

Change:

Facilitating and managing improvement in all aspects of the enterprise.

Another important reason to consider EA adoption is the *need for an organization to stay committed to its long-term goals*. The agility of an enterprise is dependent on a long-term implementation strategy using EA while short-term implementation creates a temporary illusion of an agile enterprise.

Therefore, EA is a mechanism to help her adopters remain focused on the achievement of long-term visions while providing a framework for managing everyday operational risks. To respond to the constant changes in business needs, a stable platform is needed to support enterprise operations.

The traditional approach to building an information system, by purchasing applications specifically for a department or a unit area; increases complexity, introduces redundancy, and hinders the enterprise from growing. This knows as business silos. It is whereby individually the application functions effectively but when combine gives no foundation for execution of enterprise processes.

However, the introduction of EA into an enterprise process is a holistic approach taken to address the organization-wide application needs. With, a clear understanding of how each component relates to others both at the data, software, and hardware levels of abstraction resulting in integrated silos architecture.

c. Unified Architecture Framework

The Unified Architecture Framework (UAF) is an enterprise architecture framework that provides a comprehensive and integrated view of the architecture of a system or organization. It is based on the UML (Unified Modeling Language) standard and supports modeling of systems at different levels of abstraction, including operational, system, and software levels. UAF includes a set of viewpoints that represent different perspectives of the system, such as operational, system, software, hardware, and security. It also includes a set of architecture patterns and best practices that can be used to guide the development of architectures for complex systems. UAF provides a flexible and adaptable framework that can be used across different domains and industries. It's a complex but flexible enterprise architecture framework suitable for military and government software development as well as use in commercial businesses.