UNIT – 2

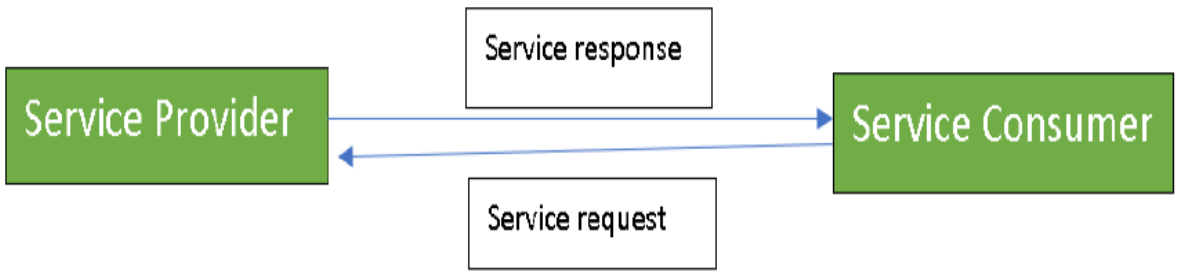
Service-Oriented Architecture-

Service-Oriented Architecture (SOA) is an architectural approach in whichapplications make use of services available in the network. It organizes a software system into a collection of interacting services. In this architecture, services are provided to form applications, through a communication call over the internet.

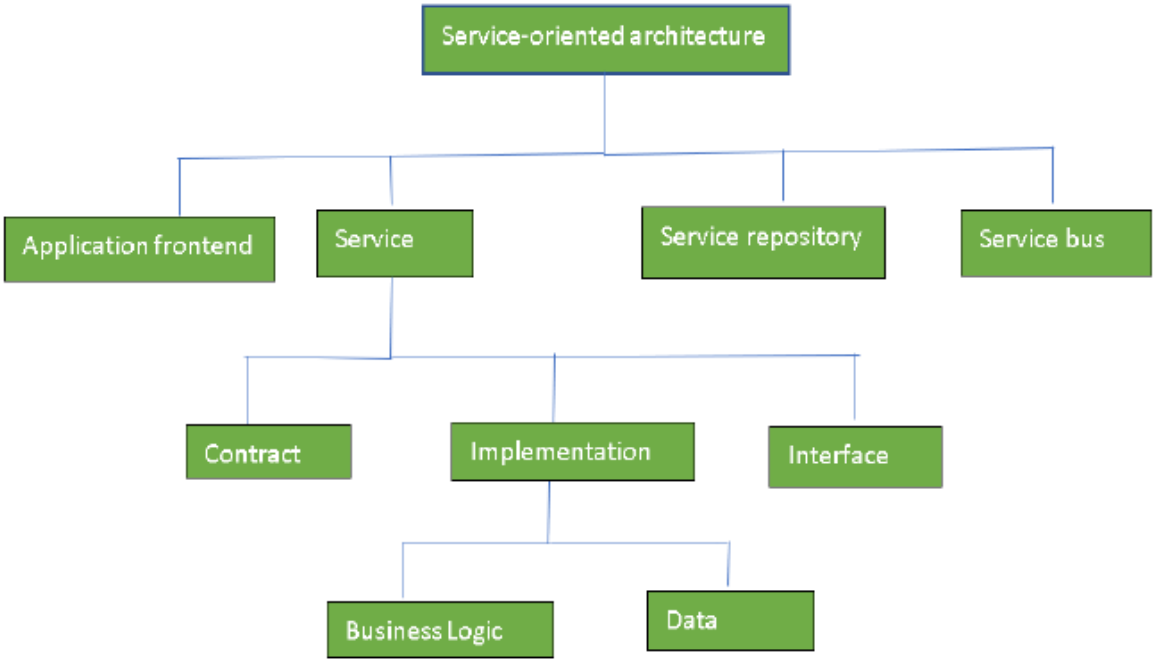
- SOA allows users to combine a large number of facilities from existing services to form applications.
- SOA encompasses a set of design principles that structure system development and provide means for integrating components into a coherent and decentralized system.
- SOA based computing packages functionalities into a set of interoperable services, which can be integrated into different software systems belonging to separate business domains.

There are two major roles within Service-oriented Architecture:

1. **Service provider:** The service provider is the maintainer of the service and theorganization that makes available one or more services for others to use. To advertise services, the provider can publish them in a registry, together with a service contract that specifies the nature of the service, how to use it, the requirements for the service, and the fees charged.
2. **Service consumer:** The service consumer can locate the service metadata in theregistry and develop the required client components to bind and use the service.



Components of SOA:



Guiding Principles of SOA:

1. **Standardized service contract:** Specified through one or more servicedescription documents.
2. **Loose coupling:** Services are designed as self-contained components, maintainrelationships that minimize dependencies on other services.
3. **Abstraction:** A service is completely defined by service contracts and descriptiondocuments. They hide their logic, which is encapsulated within their implementation.
4. **Reusability:** Designed as components, services can be reused more effectively,thus reducing development time and the associated costs.
5. **Autonomy:** Services have control over the logic they encapsulate and, from aservice consumer point of view, there is no need to know about their implementation.

6. **Discoverability:** Services are defined by description documents that constitutesupplemental metadata through which they can be effectively discovered. Service discovery provides an effective means for utilizing third-party resources.
7. **Composability:** Using services as building blocks, sophisticated and complexoperations can be implemented. Service orchestration and choreography provide a solid support for composing services and achieving business goals.

Advantages of SOA:

- **Service reusability:** In SOA, applications are made from existing services.Thus,services can be reused to make many applications.
- **Easy maintenance:** As services are independent of each other they can beupdated and modified easily without affecting other services.
- **Platform independant:** SOA allows making a complex application by combiningservices picked from different sources, independent of the platform.
- **Availability:** SOA facilities are easily available to anyone on request.
- **Reliability:** SOA applications are more reliable because it is easy to debug smallservices rather than huge codes
- **Scalability:** Services can run on different servers within an environment, thisincreases scalability

Disadvantages of SOA:

- **High overhead:** A validation of input parameters of services is done wheneverservices interact this decreases performance as it increases load and response time.
- **High investment:** A huge initial investment is required for SOA.
- **Complex service management:** When services interact they exchange messagesto tasks. the number of messages may go in millions. It becomes a cumbersome task to handle a large number of messages.

Practical applications of SOA:

SOA: -SOA is used in many ways around us whether it is mentioned or not.

1. SOA infrastructure is used by many armies and air force to deploy situational awareness systems.
2. SOA is used to improve the healthcare delivery.

3. Nowadays many apps are games and they use inbuilt functions to run. For example, an app might need GPS so it uses inbuilt GPS functions of the device. This is SOA in mobile solutions.
4. SOA helps maintain museums a virtualized storage pool for their information and content

SOA can be realized through several technologies. The first implementations of SOA have distributed object programming technologies such as CORBA and DCOM. In particular, CORBA has been a suitable platform for realizing SOA systems because it adopts interoperability among different implementations and has been designed as a specification supporting the development of industrial applications. Nowadays, SOA is mostly realized through Web services technology, which provides an interoperable platform for connecting systems and applications.

Web Services

Different books and different organizations provide different definitions toWeb Services. Some of them are listed here.

- A web service is any piece of software that makes itself available over the internet and uses a standardized XML messaging system. XML is used to encode all communications to a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. As all communication is in XML, web services are not tied to any one operating system or programming language—Java can talk with Perl; Windows applications can talk with Unix applications.
- Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML.
- Web services are XML-based information exchange systems that use the Internet for direct application-to-application interaction. These systems can include programs, objects, messages, or documents.
- A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.To summarize, a complete web service is, therefore, any service that −

- Is available over the Internet or private (intranet) networks
- Uses a standardized XML messaging system
- Is not tied to any one operating system or programming language
- Is self-describing via a common XML grammar
- Is discoverable via a simple find mechanism

Components of Web Services

The basic web services platform is XML + HTTP. All the standard web services work using the following components −

- SOAP (Simple Object Access Protocol)
- UDDI (Universal Description, Discovery and Integration)
- WSDL (Web Services Description Language)

How Does a Web Service Work?

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of −

- XML to tag the data
- SOAP to transfer a message
- WSDL to describe the availability of service.

We can build a Java-based web service that is accessible from our Visual Basic program that runs on Windows.

We can also use C# to build new web services on Windows that can be invoked from your web application that is based on Java Server Pages (JSP) and runs on Linux.

Example

Consider a simple account-management and order processing system. The accounting personnel use a client application built with Visual Basic or JSP to create new accounts and enter new customer orders.

The processing logic for this system is written in Java and resides on machine, which also interacts with a database to store information.

The steps to perform this operation are as follows −

- The client program bundles the account registration information into a SOAP message.
- This SOAP message is sent to the web service as the body of an HTTP POST request.
- The web service unpacks the SOAP request and converts it into a command that the application can understand.
- The application processes the information as required and responds with a new unique account number for that customer.
- Next, the web service packages the response into another SOAP message, which it sends back to the client program in response to its HTTP request.
- The client program unpacks the SOAP message to obtain the results of the account registration process.

REST (Representational State Transfer)

REST (Representational State Transfer) is an architectural style fordeveloping web services. REST is popular due to its simplicity and the fact that it builds upon existing systems and features of the internet's Hypertext Transfer Protocol (HTTP) in order to achieve its objectives, like creating new standards, frameworks and technologies.

Advantages of REST

A primary benefit of using REST, both from a client and server's perspective, is REST-based interactions happen using constructs that are familiar to anyone who is familiarized to using the internet's HTTP.

An example of this arrangement is REST-based interactions all communicate their status using standard HTTP status codes. So, a 404 means a requested resource wasn't found; a 401 code means the request wasn't authorized; a 200 code means everything is OK; and a 500 means there was an unrecoverable application error on the server.

Similarly, details such as encryption and data transport integrity are solved not by adding new frameworks or technologies, but instead by relying on well-known Secure Sockets Layer (SSL) encryption and Transport Layer Security (TLS). So, the entire REST architecture is built upon concepts with which most developers are already familiar.

REST is also a language-independent architectural style. REST-based applications can be written using any language, Java, .NET, or JavaScript. As long as a programming language can make web-based requests using HTTP, it is possible for that language to be used to invoke a RESTful API or web service. Similarly, RESTful web services can be written using any language, so developers tasked with implementing such services can choose technologies that work best for their situation.

The other benefit of using REST is its extensive. On the server side, there are a variety of REST-based frameworks for helping developers create RESTful web services, including REST let and Apache CXF. From the client side, all of the new JavaScript frameworks, such as JQuery, Node.js, Angular and Ember JS, all have standard libraries built into their APIs that make invoking RESTful web services and consuming the XML- or JSON-based data.

Disadvantages of REST

The benefit of REST using HTTP constructs also creates restrictions, however. Many of the limitations of HTTP likewise turn into shortcomings of the REST architectural style. For example, HTTP does not store state-based information between request-response cycles, which means REST-based applications must be stateless and any state management tasks must be performed by the client.

Similarly, since HTTP doesn't have any mechanism to send push notifications from the server to the client, it is difficult to implement any type of services where the server updates the client without the use of client-side polling of the server or some other type of web hook.