On up to two pages, give an answer to the following questions, in text and diagrams, as

appropriate:

* What is SOA?

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

Source: <https://www.ibm.com/in-en/cloud/learn/soa>

• Which are the main building blocks of SOA architecture?

The most basic construct or building block of SOA is **a service**. Software engineering over the years has evolved from procedural to structured programming to object-oriented programming to component-based development and now to service oriented.

Source:<https://www.informit.com/articles/article.aspx?p=1663690#:~:text=The%20most%20basic%20construct%20or,and%20now%20to%20service%20oriented>

• What are the advantages and disadvantages of applying it in enterprise applications

development?

Advantages

1. Maintenance is Easy – Editing and updating any service implemented under SOA architecture is easy. You don’t need to update your system. A third party maintains the service, and any amendment in this service won’t have an effect on your system. In most cases, the previous API work because it is functioning before.

2. Quality of Code Improved – As services run freelance of our system, they have their own variety of code; therefore, our code is prevented from redundancy. Also, our code becomes error-free.

3. Platform Independence – Services communicate with alternative applications through a common language, which implies it’s freelance of the platform on which that application is running. Services can provide API in different languages, e.g. PHP, JavaScript, etc.

4. Scalable – If any service obtains several users, it is often simply scalable by attaching additional servers. This will create service out there all time to the users.

5. Reliable – Services square measure typically tiny size as compared to the full-fledged application. So it’s easier to correct and check the freelance services.

6. Same Directory Structure – Services have an equivalent directory structure so customers can access the service information from an equivalent directory on every occasion. If any service has modified its location, then the additional directory remains the same. This is very helpful for consumers.

7. Independent of Other Services – Services generated using SOA principles are independent of each other. So services are often utilized by multiple applications at an equivalent time.

Disadvantages

1. High Bandwidth Server – Therefore, net service sends and receives messages and knowledge often times, so it simply reaches high requests per day. So it involves a high-speed server with plenty of information measures to run an internet service.

2. Extra Overload – In SOA, all inputs square measure its validity before it’s sent to the service. If you are victimization multiple services, then it’ll overload your system with further computation.

3. High Cost – It is expensive in terms of human resources, development, and technology.

Source: <https://www.educba.com/what-is-soa/>

• Which basic principles should the developers of SOA consider?

* Standardized Service Contract. Services adhere to a service-description.
* Loose Coupling. Services minimize dependencies on each other.
* Service Abstraction.
* Service Reusability.
* Service Autonomy.
* Service Statelessness.
* Service Discoverability.
* Service Composability.

Source: <https://www.xenonstack.com/insights/service-oriented-architecture>

There exists a document, known as SOA Manifesto, which defines six values of priority

in the work of the developers of SOA. Write those values in your own priority order

(these you find most important should be at the top of the list) and explain your

arguments.

Im not sure what this priorities entail but I think business value is the most important one followed by Flexibility since this is the primary reason behind using SOA.

**Business value** over technical strategy  
**Strategic goals** over project-specific benefits  
**Intrinsic interoperability** over custom integration  
**Shared services** over specific-purpose implementations  
**Flexibility** over optimization  
**Evolutionary refinement** over pursuit of initial perfection