Patrón cliente-servidor
Un servidor provee de múltiples servicios a múltiples clientes. Los clientes solicitan servicios del servidor el cual los provee con los servicios relevantes. Esto continua en cuanto los clientes lo necesiten.

Patrón maestro-esclavo
Aquí juegan un maestro y un esclavo, donde el maestro distribuye el trabajo entre los esclavos y consigue los resultados de cada uno para computar el resultado final.

Patrón de filtro de tubería
Se utiliza para estructurar sistemas que producen y procesan flujos de datos. Cada paso de procesamiento se encapsula en un filtro. Todo dato a ser procesado se pasa por “tubos” que se utilizan para sincronizar o a manera de buffer.

Patrón de intermediario
Este patrón es para estructurar sistemas distribuidos con componentes desacoplados. Estos componentes pueden interactuar entre ellos con servicios de invocaciones remotos. Un componente intermediario es responsable para coordinar la comunicación entre componentes.

Patrón de igual a igual
Se tienen componentes individuales que se conocen como iguales. Los iguales pueden ser servidores o clientes, solicitando servicios de otros iguales o proveyendo servicios a otros iguales. Estos pueden cambiar sus roles dinámicamente con el tiempo.

Patrón de bus de evento
Este patrón maneja eventos con 4 componentes: un event source, un event listener, un canal y un event bus. Los event listeners se suscriben a canales particulares y son notificados de mensajes que son publicados en los canales a los que se suscribieron.

Modelo-vista-controlador
Divide la aplicación en 3 partes, un modelo que contiene la funcionalidad y los datos, una vista o view que contiene la información para el usuario, y un controlador que maneja los inputs del usuario. La separación interna de la información y de su manejo permite que se cree un código eficiente y reusable.

Arquitectura orientada a servicios (SOA)
SOA por sus siglas en inglés (Software Oriented Architecture) es un estilo arquitectural para el desarrollo de aplicaciones de software que utiliza servicios disponibles en una red (como por ejemplo, la web). Promueve el loose coupling (acoplamiento flexible o suelto) entre componentes de software para que puedan ser reutilizados. Las aplicaciones construidas con SOA se basan en servicios; implementaciones de una lógica de negocio bien definida. Esos servicios pueden ser consumidos por clientes en diferentes aplicaciones o procesos de negocio.

Arquitectura basada en microservicios (MSA)

Esta arquitectura estructura aplicaciones como una colección de servicios que son altamente mantenibles y testeables, utilizan el loose coupling, son independientemente desplegables, se organizan alrededor de capacidades de negocio y pertenecen a un equipo pequeño. Estos servicios se consideran microservicios y se utilizan otros patrones que encajan con esta arquitectura para poder permitir una entrega rápida, frecuente y confiable de aplicaciones complejas. También promueve una organización que permite que tecnologías se apilen.