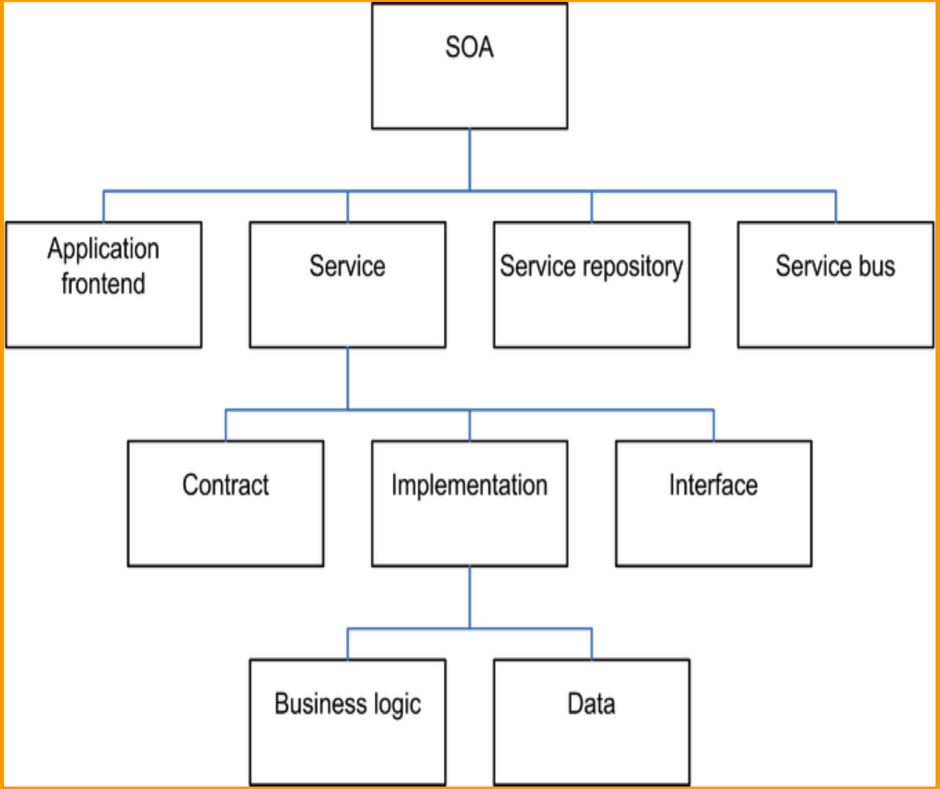
MicroServices

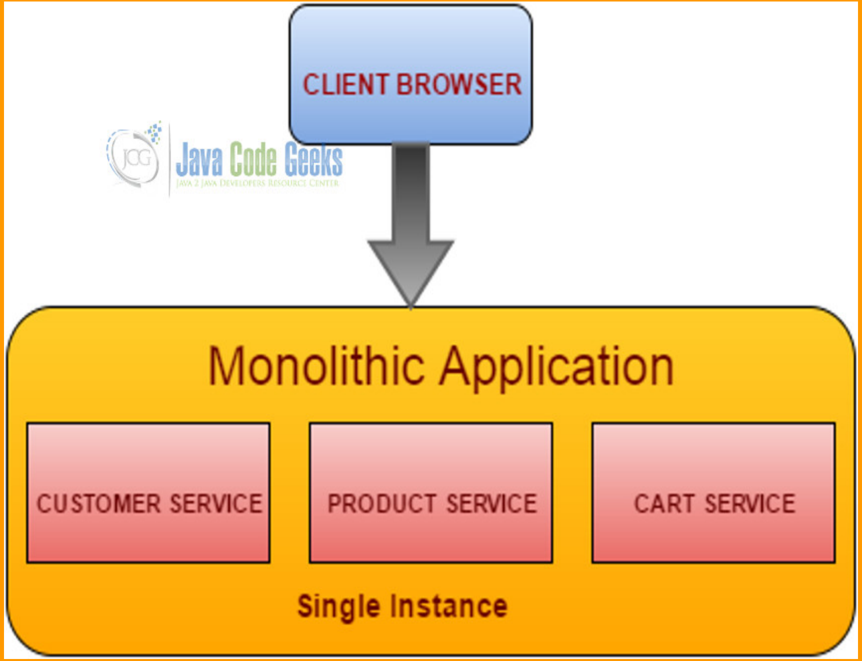
Group 11

SOA(Service Oriented Architecture)

- **Business value** is given more importance than technical strategy.
- **Strategic goals** are given more importance than project-specific benefits.
- **Intrinsic interoperability** is given more importance than custom integration.
- **Shared services** are given more importance than specific-purpose implementations.
- **Flexibility** is given more importance than optimization.
- **Evolutionary refinement** is given more importance than pursuit of initial perfection.



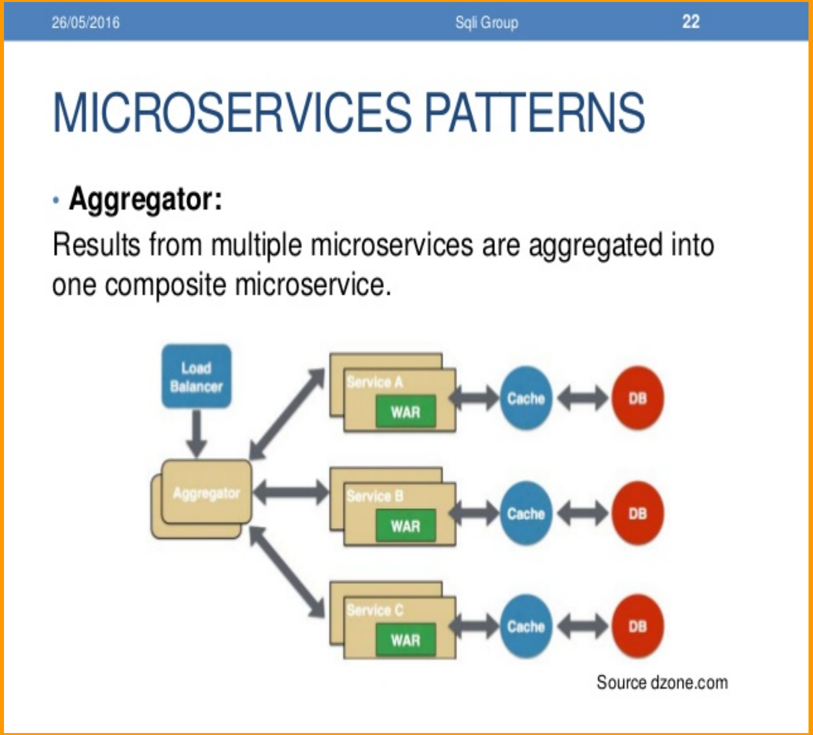
Monolithic Apps



- End-to-End functionality
- Independent from other services
- Usually not enhanced, instead rewritten
- Most likely following FRD/BRD project structure
- Single deployed unit
- Easily handled by a Central operations team

MicroServices

- Single function per service
- Easily testable, usually through automation e2e testing.
- Seamless to patch and deploy independently without having to restart the entire app/ other services
- Can easily aggregate multiple services for business functionality



26/05/2016 Sqli Group 22

MICROSERVICES PATTERNS

- **Aggregator:**

Results from multiple microservices are aggregated into one composite microservice.

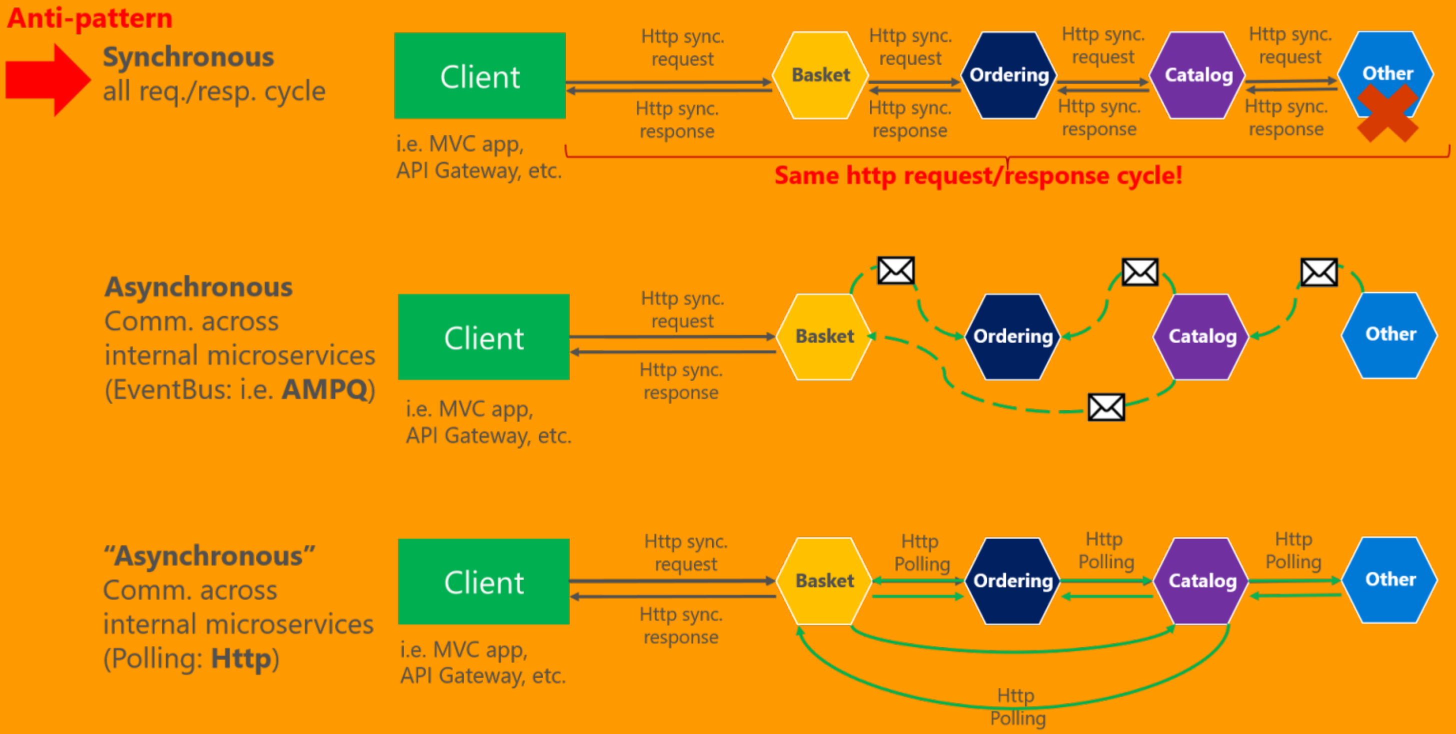
Load Balancer
Aggregator
Service A
WAR
Cache
DB
Service B
WAR
Cache
DB
Service C
WAR
Cache
DB

Source dzone.com

Communication Methods

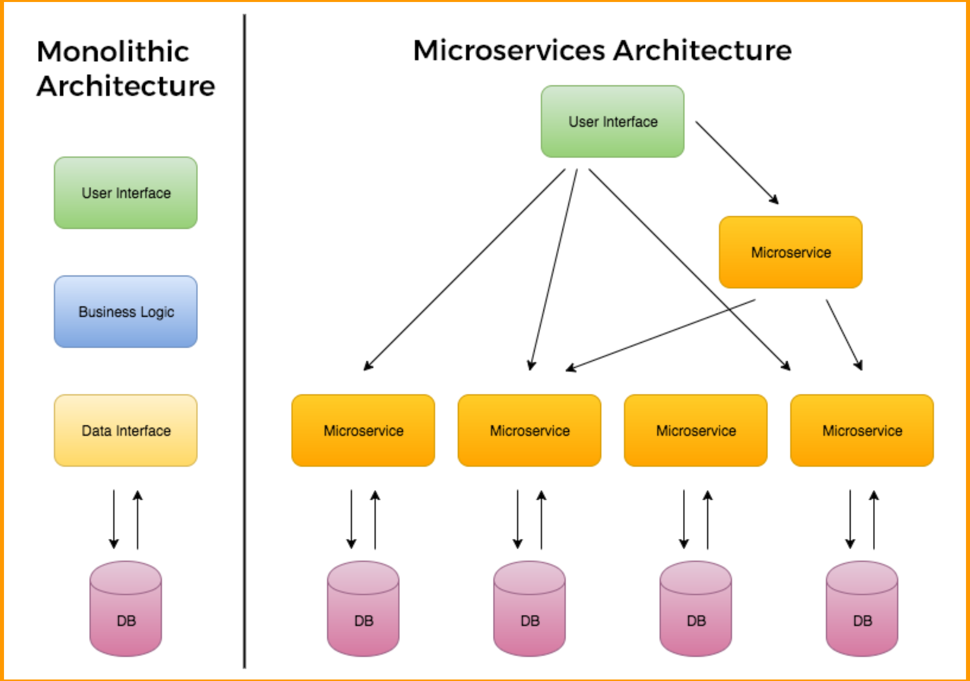
- Each service instance is typically a process, therefore services must interact using an inter-process communication protocol: HTTP, AMQP, TCP
- **Synchronous** Protocol: Client code can only continue its task when it receives a response. Example HTTP
- **Asynchronous** Protocol: Client code or message sender usually does not wait for a response Example AMQP
- Communication can have either a **single** or **multiple** receivers
  - Single Receiver: Each request must be processed by exactly one receiver or service. Example Command pattern
  - Multiple receivers: Each request must be processes by zero to multiple receivers (must be

Synchronous vs. async communication across microservices



Scalability

- Scale only those services that need scaling as load demand increases
- Docker, Kunernetes
- Deploy each service instance as a container


Monolithic Architecture
User Interface
Business Logic
Data Interface
DB
Microservices Architecture
User Interface
Microservice
Microservice
Microservice
Microservice
Microservice
DB
DB
DB
DB