1. 接入层：

api网关：处理南北流量，也就是外部用户请求(http或者tcp)。统一处理通用逻辑：黑白名单，验签，鉴权，流控，协议转换，设置通用上下文（threadlocal），服务编排（调用对应服务逻辑），结果转换

目前成熟框架：spring cloud gateway。Zull2

1. 服务治理

注册中心：实现服务注册和服务发现。目前有cp和ap等一致性模式

成熟框架：zk，nacos。Eureka，consul等

rpc框架：实现高性能远程调用，客户端软负载均衡调用

成熟框架：dubbo，spring cloud，grpc，motan，

分布式配置中心：配置的分布式存储和实时生效。下发等功能。可灰度，可回滚

成熟框架：git，spring cloud config，apollo

链路监控和跟踪：服务的全链路监控，可对整体链路进行分析，包括链路上下文传递以及traceid的监控，瓶颈分析

成熟框架：brave+zipkin，cat

服务熔断降级：对服务异常情况下的保护和自愈，可以在服务出现超时，异常等情况下设置熔断或者有损服务策略

成熟框架：hystrix，sentinel，

服务流控：在流量过载以后实现对流量进行控制，保护服务提供正常的服务能力

成熟框架：guava limiter（本地），sentinel（本地和分布式），redis（分布式）

故障演练：针对于服务的日常故障演练和注入，配置平台

成熟框架：chaosblade

Servicemesh（大趋势）：

服务网格。实现灰度流量。流量转发。负载均衡。限流，日志，全链路等功能合计

成熟框架：istio，mosn

1. 存储层：

关系型数据库层面（mysql）：核心业务数据存储和查询

高并发和海量数据情况下：

数据需要拆分和实现路由规则功能，实现水平拆分（分库分表）和垂直拆分（业务分库）

数据拆分：

客户端实现和proxy实现

客户端实现：sharding-sphere（sharding-jdbc）

proxy实现：atlas等

唯一主键生成：在分库分表以后需要唯一主键生成组件：snowflake（雪花算法）

异构数据同步：需要通过binlog等手段把关系型数据同步到搜索引擎或者大数据组件

成熟框架：canal或者databus

部署结构：

主从结构（通过binlog复制），客户端支持读写分离

缓存

分布式缓存（Redis）：提供高性能的读取数据能力

不同的数据结构（kv，hashmap，set，zset，list，bitsit）

高可用：redis-cluster，哨兵模式

关注点：大key，热key，缓存雪崩等问题（本地缓存兜底）

使用场景：缓存，分布式锁（setnx），注册中心，队列，限流

本地缓存：guava cache，caffeine cache

主要是针对不经常变化的内存数据进行存储

消息队列：用于系统的解耦和流量缓冲

Kafka：目前主要用在日志传输。大数据分析，系统解耦等各种场景

Rocketmq：阿里开源，支持更多的业务场景，消息指定tag，分布式消息，消息轨迹等

其他部分：

搜索引擎存储：elasticsearch和solr

支持结构化和非结构化的存储和搜索。支持分词，联想等高阶搜索。补充关系型数据库的一些查询方面的劣势，使用倒排索引（lucene）

列式存储：hbase，Cassandra

针对于大数据存储的场景。尤其是类似于im聊天记录，feed流，traceid等，内部实现是lsm树

文件存储：

主要是存储图片或者超大文件，比如oss。支持秘钥加密和解密

1. 监控：

监控：大型互联网系统必备。包括系统级别监控和业务级别监控

主要框架：

系统级别：metric+influxdb+grafana

日志级别：elk（elasticsearch+logstash（filebeat）+kibana）

全链路监控：zipkin

1. 服务部署和编排

目前主流容器化：k8s+docker

针对于服务的部署和编排目前主要实现的k8s+docker。通过k8s的抽象几个层级：

Pod，deployment，service，ingress，gateway等。同时支持部署单元的扩容缩容

，故障节点替换等

单元化部署：按照流量进行整体单元部署（zone）

异地多活：

1. 大数据相关（数据处理和分析，机器学习，推荐等场景）

存储：hadoop，hdfs

实时计算：jstorm，flink

离线计算：mapreduce，spark

1. 安全

加密算法，oath，sso（jwt），双因子认证