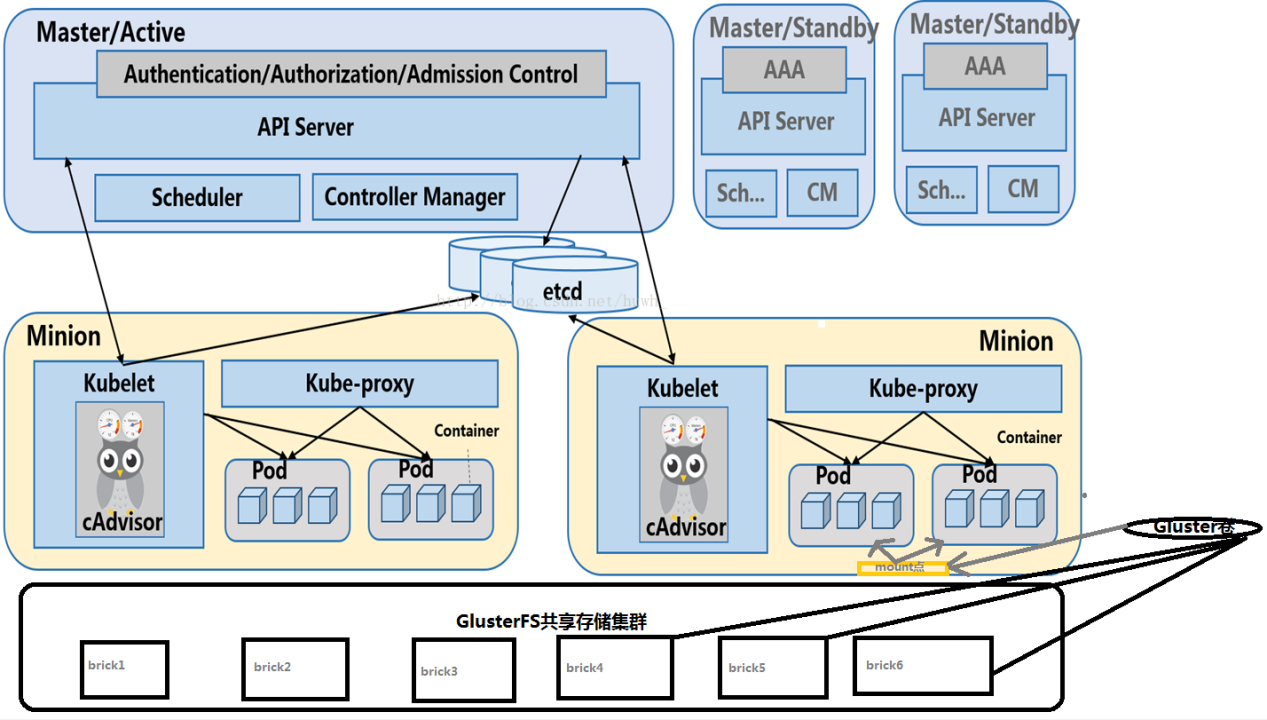
## 使用k8s优势分析

1. **系统介绍**

**说明：**Kubernetes是Google开源的一个容器编排引擎，它支持自动化部署、大规模可伸缩、应用容器化维护和管理。生产环境中，我们使用Kubernetes创建多个容器，每个容器里面运行一个应用实例，然后通过内置的负载均衡策略，实现对一组应用实例的管理、发现、访问，而这些细节都不需要运维人员去进行复杂的手工配置和处理，这将极大减轻运维人员的负担和运维成本。



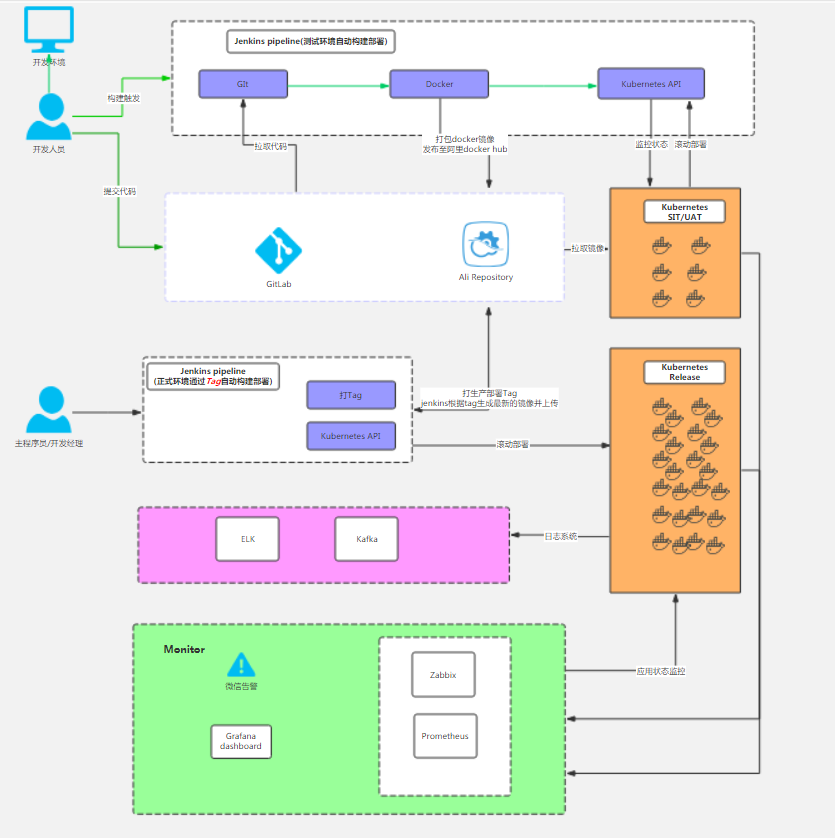
图一、k8s核心架构图

1. **优势分析**
2. **提高资源利用率，资源集中且利于分配；**

每个服务系统以pod为单元运行于k8s上，k8s调度器会按照每个Node的实际硬件资源情况对pod进行随机调度，k8s将自动对CPU和内存和硬盘资源进行更加高度的规划。比如内存使用上，基于容器可靠的资源隔离性，可以对某个pod进行最大资源限定（一般pod初始时设为4G），那么一台Node节点最多就可以运行15-20个pod之多，这样就充分地利用了物理机64G的内存空间（CPU同理）；再比如硬盘资源使用上，由于物理部署时会在每个物理机上至少挂载一块硬盘，使用k8s后，只需要在k8s集群机器上挂载一块硬盘供各个节点使用即可，业务系统均采用GlusterFS共享存储进行业务数据持久化，这样就可以大大提高硬盘的使用率。

1. **提高系统管理的效率，进行快速发布和更新服务；**

运维通过Git+Maven+Jenkins+Harbor搭建一键式上线流程，让应用发布、更新、回滚等操作变得简单快捷，可以极大地减低运维成本。



图二：上线流程图

1. **提高系统的可靠性，系统及时发现宕机服务并自动重启；**

业务系统时常出现假死的情况（端口存活但停止工作），k8s通过业务系统提供的判活接口可定期向pod实例发送请求，多次请求失败后可认定系统宕机，k8s就会立马对宕机或者假死的服务进行重启操作，整个过程完全无需人工参与。（我们已规定各研发组必须为各个服务提供基于http协议的判活接口）

1. **易于扩展、横向扩展；**

当系统发现某个服务pod压力过大时，可对该服务pod自动进行副本数量调整，压力恢复正常时自动杀死多余副本，从而实现服务的自动伸缩（基于k8s的HPA机制）；当整个集群的硬件资源吃紧时，可以通过增加Node节点，平滑地将集群规模进行扩容；

1. **开发、测试、生产环境统一，一个镜像到处运行；**

开发、测试、运维在同一套环境中调试、上线服务，开发人员只需要将自己的代码打成镜像便可在任意k8s环境中运行（前提：打通特定网络路由），不用再担心因物理环境的差异带来的困扰。

1. **各个服务拥有更加完善的隔离空间（网络，端口，硬件资源等）**

各个服务在pod中运行，拥有自己独立的虚拟IP和服务端口（网络空间隔离），彻底摆脱物理部署中多个服务端口重复、冲突的问题；k8s通过资源限定声明，可将硬件资源严格划分和隔离，不应担心因某个服务内存泄漏或资源吞噬影响其他服务正常运行，系统安全性和稳定性得到了极大提升。

1. **系统健康检测、日志访问、系统监控等功能模块化、插件化，功能体系更加完善；**

系统通过健康检查端口可及时发现假死应用；服务日志通过log-pilot->kafka->ES/Flume进行实时查看和异地存储；监控系统采用Prometheus+grafana，做到各种粒度的监控（Node、Pod、Container、JVM等等）。整个系统各个功能组件完全解耦，程模块化分布，易于管理与维护。