容器云建设方案

版本记录

版本	文件名	修改人	版本日期	备注
1				

目　录

第1章 项目需求分析

1.1 项目建设背景

在当今互联网的浪潮中，企业互联网技术发生了巨大的变化，架构模式从巨型架构，单层架构，SOA 架构到微服务、无服务器架构；开发流程从瀑布式到敏捷开发、DevOps ；部署方式从单一应用服务器到云端，从虚机到容器；基础设施从托管，自建到私有云，公有云，混合云；展示形态从 pc 端、移动端到多端化；从各类应用系统，部分公共组件到公共支撑平台，大数据平台。

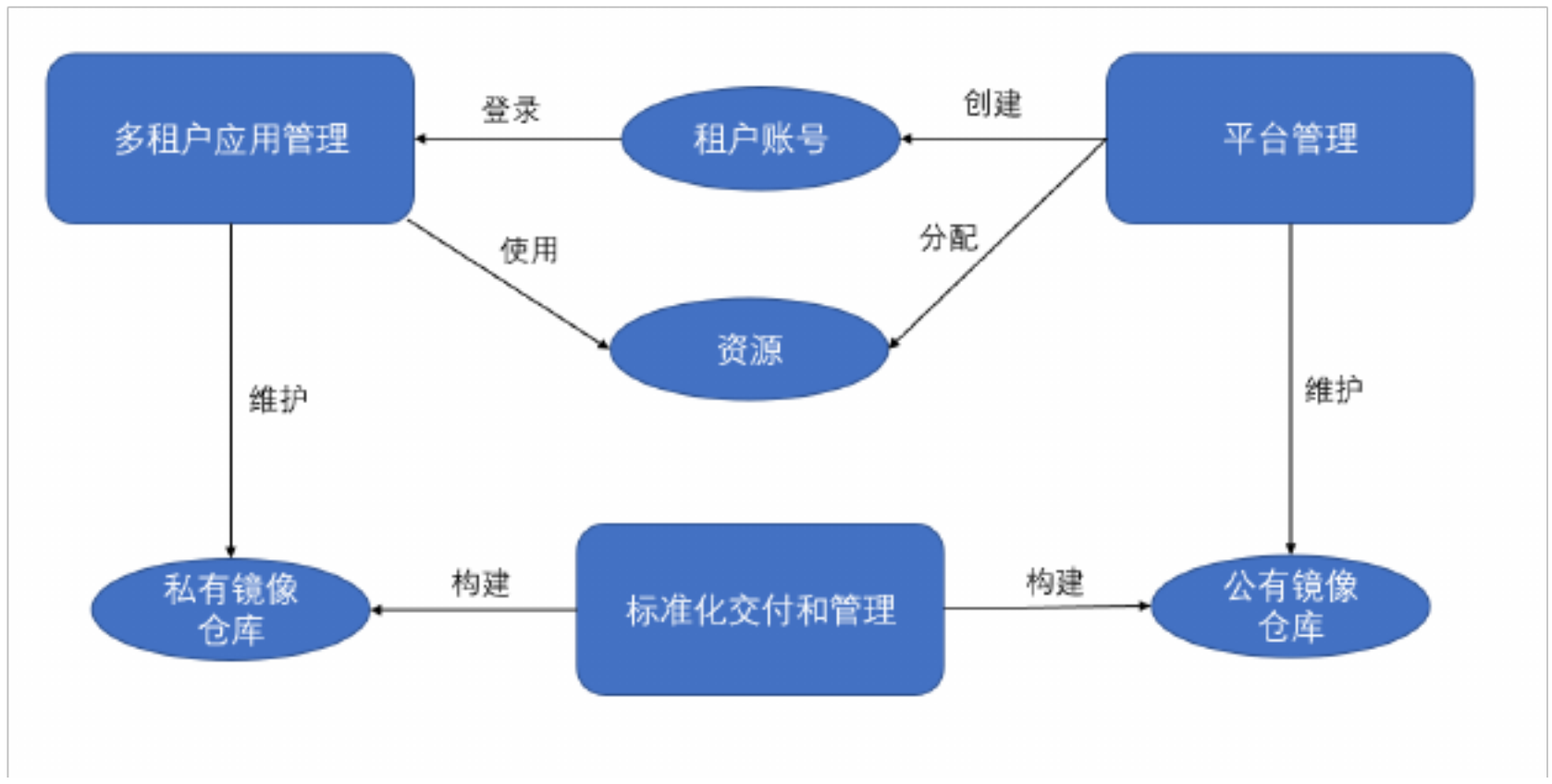
弹性伸缩是云资源高效使用的一个重要特性，也是实现业务敏捷的一个关键手段，是企业用户非常关注的应用场景。

容器云，是轻量化 PaaS 平台的一种容器化实现方式。是基于容器技术、容器调度编排技术和支撑容器运行操作系统技术、分布式技术上构建的云平台。从 PaaS 功能来说，PaaS 提供应用开发、应用托管、应用运维等能力。也就是说在 PaaS 上，客户可以开发、测试应用，用户可以托管应用到 PaaS 的运行时环境，同时 PaaS 提供给客户应用服务运维的能力（比如应用管理、监控、日志、安全等）。PaaS 平台是以应用开发为中心，以解决应用全生命周期管理、公共支撑平台服务、基础资源的高效利用。从应用的开发、部署，到运维的全流程生命周期管理，实现应用的自动伸缩、弹性扩展、灰度发布以及监控告警、故障分析、自动迁移、自动恢复；提供丰富的预集成服务,把通用的软件能力服务化，使得应用能快速拥有分布式的高可用性、高可扩展性；通过对底层资源的抽象和资源层的隔离，尽可能地共享或平摊资源，以提高资源整体使用率，从而降低基础设施的投入。

容是就是满足互联网业的新技术需求。

1.2 业务需求

根据项目整体规划，功能性需求总共分为三个部分，容器云平台管理和资源管理、多租户管理和应用管理，标准化交付和管理。各部分的关系梳理如下：



平台管理要求具备平台资源管理和资源隔离、租户账户管理、公共镜像库（中间件商店）、分层安全机制、高可用、易于扩展/更新/迁移能力。预留将来可扩展支持混合云管理等能力。重点在于资源管理，具备向租户按需提供基础设施资源的能力。平台管理同时创建租户账号，维护公共镜像仓库和仓库中镜像。

多租户特性需求要求具备租户管理、租户资源管理、租户应用管理、微服务及服务治理、私有镜像库、租户安全等能力。重点是租户应用管理及服务治理能力，以及安全能力。同时提供维护租户下用户、权限、角色、组织架构、分配的资源，租户私有镜像仓库的能力。租户账号由容器员平台管理员创建，租户使用租户账号登录容器云平台租户管理界面，维护租户下的组织架构、用户、角色、权限、资源、应用、服务等。

标准化交付的对象是镜像，向公共镜像仓库或私有镜像仓库构建镜像。公共镜像是指对所有租户可用的镜像，通常由平台管理员来维护。私有镜像是租户自己的镜像，仅对租户自身可见。同时支持镜像上传下载同步等能力。重点是构建持续集成工具链（使用 Jenkins ），持续集成止于镜像仓库，以镜像仓库为媒介实现持续部署、持续反馈、持续改进闭环 DevOps 流程。

根据每部分的详细需求，我们定义平台管理员视图和租户视图，以及持续集成流程实现标准化交付能力，详细功能如下：

平台管理	多租户特性	标准化交付和管理
平台Dashboard	租户Dashboard 租户管理 组织架构管理/人员管理/角色管理/权限管理	镜像仓库管理
资源管理与资源隔离 集群管理 分区管理 节点管理 网络管理 存储管理 标签管理	租户资源管理 租户、用户分配使用的资源 应用管理 业务应用/应用模板管理 业务服务/服务实例管理 弹性伸缩/负载均衡/健康检查 服务标签管理 链路跟踪	镜像管理 持续集成 持续部署 测试管理
租户账户管理	微服务治理 集中日志/监控告警	缺陷管理
公共镜像仓库管理	注册发现 服务配置 服务网关	文档管理
平台安全机制	私有镜像仓库管理 租户下安全机制	

平台初始化之后只有平台管理员账户，租户账户平台管理员按需创建。安全需要覆盖全方位，确保每个层次都具备可靠的安全机制。

必须建立基于 Spring Cloud 框架的微服务支持。

1.3 技术需求

容器服务（Container Service ）是一种高度可扩展的高性能容器管理服务，服务于应用的完整生命周期。通过Docker 容器来运行或编排应用程序，您将不再需要安装、运维、扩展自己的集群管理基础设施。容器服务具有简单易用、灵活弹性、秒级部署等特点，通常具备以下能力：

1.3.1应用托管

提供大规模容器集群管理、资源调度、容器编排、代码构建，屏蔽了底层基础构架的差异，简化了分布式应用的管理和运维。

1.3.2内部 DNS 服务

为每个服务提供二级域名和端口映射，服务之间可通过内网域名进行访问，不会受容器重启、迁移或扩展的影响。服务之间还可通过环境变量链接起来。

1.3.3弹性伸缩

容器服务的弹性伸缩通常于秒间对容器进行横向扩展。同时可对 CPU、内存等负载数据进行实时监控，实现全自动/半自动弹性伸缩。

1.3.4负载均衡

提供四层、七层负载均衡将流量引导、分摊到服务每个实例，并根据容器状态自动对负载均衡进行实时配置，提高应用整体可用性及吞吐量。

1.3.5应用性能监控

提供全方位的日志监控，自动搜集容器输出日志，并可保留已中断的容器的历史日志。可对容器性能作全方位实时监控。

1.3.6灰度升级和滚动更新

灰度升级是指在升级过程中，在用户无感知的情况下做到不停机，平滑的升级。灰度发布可以保证整体系统的稳定。

1.3.7容灾容错

具有独特的容器仓技术，可以保证容器实例的副本数量即使在某个主机出错的情况下也能维持不变。

1.4 非功能性需求

容器云平台性能需求中最重要的是安全性、可扩展性和稳定性以及满足业务服务部署管理的性能需求、用户操作友好性需求。以满足容器云平台升级、迁移、更新、扩展需求。

可用性：可用性是平台可靠性、稳定性等的综合体现，可靠稳定的平台可用性才高。通常可用性是评判一个平台或系统的最重要维度，可用的平台才能考虑性能等其他维度。

可扩展性：可扩展性或者弹性是容器云平台或者云平台的价值所在。容器云平台需要提供弹性的基础设施资源，用于业务应用的弹性扩展。

界面友好：在每一个用户登录后都有个工作台来总括该用户的资源分配使用、应用部署运行、系统组件运行状况等；然后通过链接可以直接跳转查看详情。各环节相关联成为一个闭环。在采用微服务架构之后，统一是服务和应用建设，不再有“项目”概念，一切皆是服务。服务编排为应用。

性能：性能分两个方面，一是平台的性能需求，另一个是服务的性能需求。平台的性能往往决定着其上部署的单个服务实例的最大性能。平台的性能往往取决于平台基础设施资源的能力。也就是平台的硬件设施配置。另一方面，服务的扩展能力，负载均衡能力也是实现高性能的重要方式。

1.5 性能指标需求

□ 开发测试效率达到每天 20 次的迭代

□应用并发峰值达到 3 万 TPS 的

□单台物理机可运行约 100 个容器实例，比虚拟机提高 10 倍左右

2章 项目总体设计

总体架构

2.1.1建设原则

建设容器云平台的目的是用来承载业务应用的，为了实现IT融合、应急响应、敏捷开发、应用交付、权限认证、弹性伸缩、异常迁移、环境一致性、高可用、网络隔离、资源配额、日志收集、健康检查、监控告警等能力；中远期目标是通过采用微服务架构、DevOps 方法论等逐步构建企业的公共支撑平台台。容器云平台的方案设计需要遵循建设容器云的目的和需求，满足以下设计原则：

高可用原则：关键核心组件，都要求高可用设计、高可用部署，保障在服务器宕机等故障情况下，应用不受影响。

□先进性原则：平台的建设所采用的技术是主流云计算技术，确保平台在一定时间内具备稳定的更新和保持先进性。

□成熟性原则： 技术选型确保先进性的技术上，采用成熟的技术，确保平台功能稳定。

□开放性和兼容性原则：标准化或通用的技术手段兼容主流的设备和系统、软件、工具等。

□可靠性原则：提供可靠的计算、存储、网络等资源，在平台、服务、应用、组件等方面实现高可用，避免单点故障，保证业务的连续性。

□可扩展性原则：采用分布式架构，支持在不更改整体架构的前提下进行系统扩容和业务范围的扩展。平台的计算、存储、网络资源等根据业务应用工作负载的需要进行动态伸缩。

□松耦合原则：容器云平台架构采用松耦合架构，平台各组件提供开放标准的API 接口，方便客户已有系统的集成或平台组件的扩展和替换。

伸缩、负载均衡、异常迁移等；纵向：协议层加密、网关、访问控制、过滤、限流、熔断、禁用 root 用户运维权限、系统资源用户受限访问、资源隔离、网络策略等。容器云平台应该在各个层面进行完善的安全防护，确保信息的安全和私密性。

□用户友好原则：注重用户体验，提供简洁便利的操作体验，避免二义性，界面操作保证流畅简单。

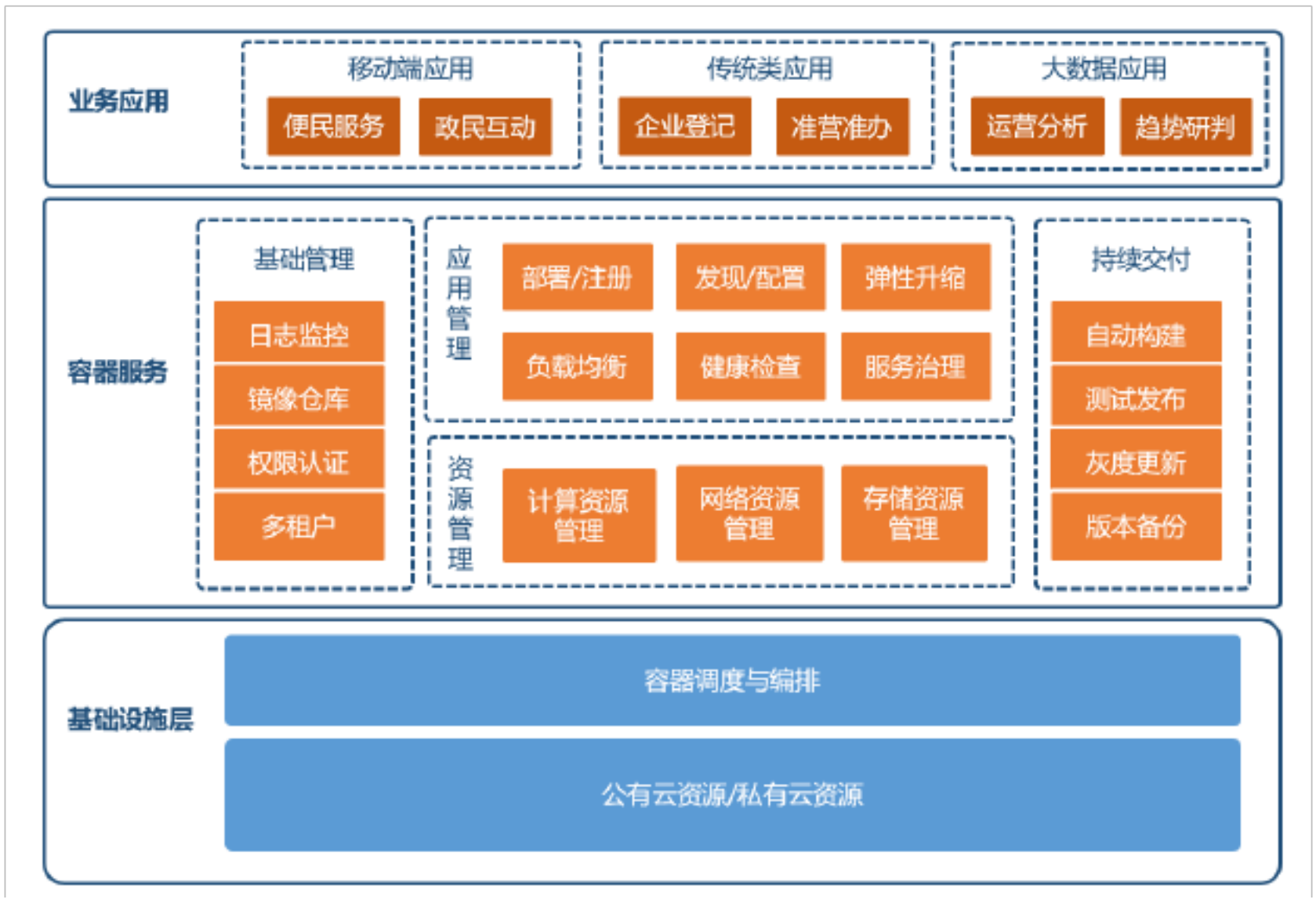
□隔离性原则：容器云平台租户之间实现隔离，租户和平台管理之间实现隔离。

□生态原则：容器云平台需要构建生态体系，提供和集成认证、权限、配置、日志、监控、告警、注册发现、网关等能力，支撑真正的企业业务应用。

2.1.2技术架构

容器云平台是以承载业务应用为中心的，其设计重点在于为业务应用的运营提供资源和分租户管理运维业务应用。同时为了提升敏捷开发和测试的能力，提升持续集成、持续反馈、持续响应的能力，实现合理的 DevOps 流程和工具选择。容器云平台架构设计时， 遵循设计原则，明确设计目标，提供真正意义上的企业级方案，权限、职责、范围要明确定义。仅有容器或容器调度管理不足以支撑业务应用，需要构建容器云生态组件体系，包括认证、权限、配置、日志、监控、告警、注册发现、网关等组件。容器云平台各生态组件松耦合部署，采用开放性标准 API 接口，可以根据需要进行扩展或替换。

平台技术架构如下图所示：

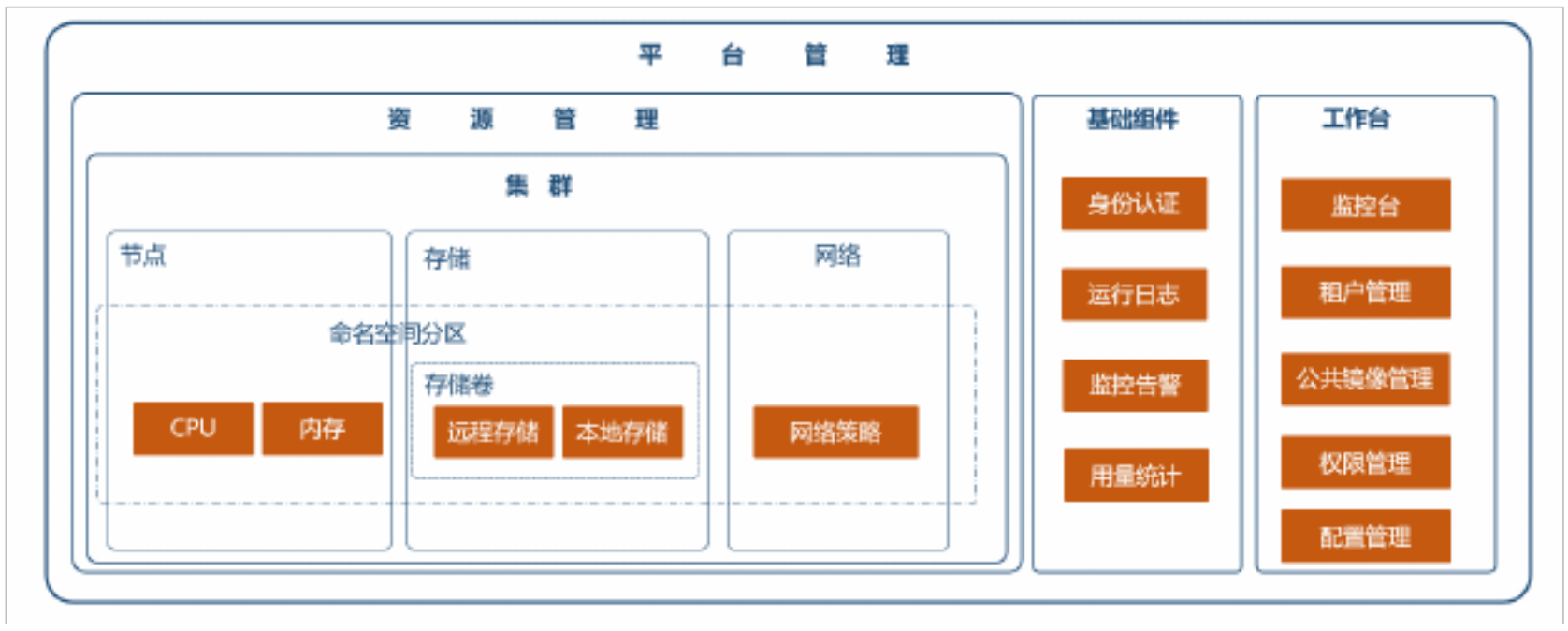


器云平台基于容器技术，通过容器编排和调度框架，实现对基础设施资源层的管理和对应用服务的资源供给，提供计算、网络、存储等资源。容器云平台提供整个平台的日志服务、监控服务、认证服务、权限服务、多租户能力、镜像仓库等基础组件。

租户管理和租户应用管理提供灵活的组织架构、用户、角色、权限的定义和业务应用、服务的运营阶段管理。CI/CD 实现业务应用、服务的开发、测试、部署发布阶段管理，平台完整实现业务应用、服务的整个生命周期管理能力。必须支持项目中微服务应用中心以及身份认证中心的上线，以及互联网应用、大数据应用等。

平台管理重点在于资源管理，资源划分为集群、节点（CPU、内存）、存储（本地、远程存储）、网络。容器云平台可能有多集群，每个集群在创建时定义集群网络，管理容器云节点，存储。一个集群可能有一到多个逻辑分区，分区是资源的逻辑划分。容器云的资源要支持标签，通过标签来分类资源，在应用/服务部署时作为资源选择的依据。根据选择的网络类型来确定是否支持网

监控告警、计量计费等基础功能，这些组件和平台是松耦合架构，但各组件之间需要实现统一认证、单点登录能力。



容器云平台 Dashboard 展示平台资源使用情况（总的资源、已使用、可用）以及资源使用最多的 top 10 租户及使用明细；平台节点情况（节点配置、节点资源占用、可用，节点上 Pods 或容器数量，资源紧张的予以不同色彩提示）；平台 Pods 、Containers 运行情况（Pods 量、状态、异常；平台组件使用的 Pods 和租户应用使用的 Pods ，两者要分开）；公共镜像统计；平台组件运行状况。

租户账户由平台管理员创建，包括租户名称、账号、密码、租户联系方式、租户资源分配。公共镜像仓库由平台管理员来维护，提供平台级的中间件镜像服务等。

系统设置定义容器云平台级配置和设置。权限管理提供不同级别的平台管理员权限，比如平台管理员为每一个集群创建集群管理员来维护其独立的容器集群。

户管理架构设计

租户管理重点在于应用管理和服务治理能力。一切围绕业务应用的运营、治理为中心。