中国光大银行

HF17-004—容器云PaaS平台建设项目

可行性分析报告

（基础建设新建项目）

单位：中国光大银行

部门：信息科技部

文件信息

文件标题	容器云 PaaS 平台建设项目可行性分析报告
项目预算编号	HF17-004
申请部门	信息科技部
发布日期	2017-06-05

修订记录

日期	版本	描述	作者
2017-04-10	V1.0	创建	解培
2017-05-18	V1.1	增加开发中心需求	陈瑶

文件审核/审批

此文件需如下审核

姓名	职务/职称	签名	签名日期

文档分发

此文档将分发至如下个人或机构

姓名/机构	职务/职称

目录

关 键 词： 容器，容器云，Docker，弹性云计算，PaaS 平台，DevOps，分布式微服务

摘　要：为推动新兴技术在银行业的应用，应对互联网金融服务对我行传统 IT 系统带来的技术冲击，中国光大银行积极探索容器云、PaaS 平台、分布式微服务框架、DevOps 相关技术，推进开展基于开源技术的容器云 PaaS 平台建设项目。容器技术能大大提高系统应对大流量冲击时的并发处理能力，大幅缩短应用交付周期，提高部署配置效率和自动化运维水平。分布式微服务框架可以有效沉淀行内数据能力和服务能力，支撑敏捷开发，应对业务需求的快速变化。秉承“统一规划，架构先行，技术创新，管理配套”的方针，通过对容器和分布式微服务前沿技术的研究和实践，充分发挥容器技术、分布式微服务技术与云计算技术融合的技术优势，建设自主、安全可控，具备持续交付能力、高资源利用率、高应用伸缩性、连续高可用的自动化应用服务框架和 IT 管理体系架构。

1 引言

1.1 编写目的

本文档通过对容器云 PaaS 平台建设项目的项目背景、目标、项目建设需求进行描述说明，并开展针对性的项目技术解决方案设计，通过对项目建设的经

济可行性、技术可行性等方面内容的论述说明，进行本项目建设的可行性研究分析，为立项管理工作所涉及各方管理职能人员提供信息输入和项目立项决策依据，并可作为后续项目建设工作的重要参照和指南。

1.2 预期读者

立项评审人员。

1.3 术语定义

- **容器** 容器技术是一种服务器资源共享方式，容器技术可以在按需部署实例的过程当中为系统管理员提供极大的灵活性。由于 hypervisor 虚拟化技术仍然存在一些性能和资源使用效率方面的问题，因此出现了一种称为容器技术（Container）的新型虚拟化技术来帮助解决这些问题。
- **容器云** 容器云以容器为资源分割和调度的基本单位，封装整个软件运行时环境，为开发者和系统管理员提供用于构建、发布和运行分布式应用的平台。当容器云专注于资源共享与隔离、容器编排与部署时，它是一种 IaaS；当容器云渗透到应用支撑与运行时环境时，是一种 PaaS 平台。
- **Docker** 是一个开源的容器引擎。
- **容器平台** 为应用开发与运维人员服务，提供支撑应用运行所需的软件运行时环境、相关工具与服务，如数据库服务、持续集成持续交付服务、日志服务、监控服务等，让应用开发者可以专注于核心业务的开发。
- **Hypervisor** 是一种运行在物理服务器和操作系统之间的中间软件层,可允

许多个操作系统和应用共享一套基础物理硬件，因此也可以看作是虚拟环境中的“元”操作系统。

- **KVM** 是一款开源的 hypervisor 软件。
- **分布式微服务** 是将原本独立的系统拆分成多个小型服务，各自围绕着系统中的某一项或一些耦合度较高的业务功能进行构建，在各自独立的进程中运行，并维护自身数据存储、业务开发、自动化测试案例以及独立部署机制，服务间使用轻量级通讯进行协作。
- **PaaS 应用平台** 为分布式微服务提供高性能服务框架、服务治理与数据化运营管理能力、云中间件服务与集成能力。

1.4 参考资料

无

2 项目背景和意义

2.1 项目必要性说明

随着移动互联网金融服务的迅速崛起，其在我行各类服务渠道中的地位愈加重要。相应的移动金融业务需求变化快、业务场景多，导致采用传统 IT 基础架构及开发、运维模式逐渐暴露出不足：竖井式的单体应用群，导致业务功能的重复开发与数据冗余。庞大的单体应用、紧耦合的业务功能、开发、测试和运维等环节被分割独立，导致项目投产上线周期相对较长，无法及时满足互联

网时代的客户和业务需求；新产品投产和业务量不断增加，导致机房投入的服务器数量和运维成本大幅提高；生产测试环境的隔离增加了相应开发、测试和运维工作人员的工作量。特别是在应对“红包”、“秒杀”抢购等高并发特殊场景时，传统的基础架构和运维方式很难适应。

以 Docker 为代表的容器技术和分布式微服务框架越来越受到银行 IT 业的关注。容器技术最显著的特点是容器中不仅包含了应用本身，还包括了应用运行所需的环境，从而能够在不同的环境中迁移并运行，做到一次构建，多次部署。另外，容器技术也是一种相对轻量级的虚拟化技术，能够做到秒级启动和停止。基于这两点主要特性，容器技术有着解决传统金融行业 IT 所面临问题的先天性优势。分布式微服务框架将庞大的应用拆分为完全解耦的微服务，服务间通过轻量级通讯进行交互。应用的解耦可以有效沉淀行内业务能力和服务能力，避免各应用对公共服务的重复开发和对公共数据的冗余存储。同时，应用微服务化，可有效缩减业务需求变化带来的应用影响范围，提升开发测试效率，基于容器和微服务技术打造的应用开发流水线，可打通我行开发测试运维全流程，支持敏捷开发。通过与容器技术相结合，PaaS 应用平台可以根据各个服务的应用级指标，进行秒级弹性伸缩等数据化治理，达到行内资源利用率最大化。

容器技术与分布式微服务技术的出现为传统金融业 IT 技术转型提供了一种新的转型思路。而且在大规模集群管理、混合云平台构建、微服务化应用以及

云原生应用等场景中，这两项技术也得到了广泛应用，很大程度上与我行 IT 的战略转型相契合。研究容器技术、分布式微服务技术及其和 IaaS 云平台融合技术，打通 PaaS 应用云服务、容器和 IaaS 云平台资源编排的界限，构建银行容器云 PaaS 平台，释放服务+容器+云平台 1+1+1 大于 3 的技术红利，为加快我行金融服务创新步伐，提高金融服务创新质量打下坚实的技术基础。

2.1.1 现状分析

2016 年银监会发布《中国银行业信息科技“十三五”发展规划监管指导意见》，指出，新一轮科技革命蓄势待发，以互联网、大数据、云计算等为代表的新兴技术与传统金融加速融合，既为银行业改革发展转型带来了新的动力，也对银行业的传统优势领域形成一定压力。银行业需要积极面对新兴技术带来的机遇与挑战，主动开展架构转型，建立开放、弹性、高效、安全的新一代银行系统，深化信息科技治理成效，完善科技研发运维体系，强化信息安全和风险管理。《指导意见》指出，银行业应该适应互联网环境下计算资源弹性变化和快速部署等需求，开展云计算架构规划，制定云计算应用策略。探索构建私有云平台，采用成熟度高、开放性强的计算虚拟化、容器虚拟化、分布式存储、网络虚拟化等技术，建立资源池，形成资源弹性供给、灵活调度和动态计量的私有云平台。探索建立银行业金融公共服务行业云，构建私有云与行业云相结合的混合云应用。同步开展应用架构规划，构建与云计算基础设施相适应的应

用架构，自主设计或推动应用开发商实施应用架构改造，并降低应用与基础架构的耦合度。稳步实施架构迁移，到“十三五”末期，面向互联网场景的主要信息系统尽可能迁移至云计算架构平台。在银行信息系统应用架构的改造升级中，容器技术成为关键，银行客户需要将现有的应用架构从大而全的 SOA 体系向更为灵活的微服务架构转变，基于容器技术和分布式微服务框架打造一个高可用、高扩展、高性能、高伸缩性及高安全的业务平台。

2.1.2 条件和约束

选用基于安全可控容器及容器编排技术、分布式微服务框架及服务治理技术进行自主研发，我行具有自主知识产权。项目在我行全行私有云范围内使用，平台内各组件均基于主流开源软件模块，整体技术成熟度较高，整体市场活跃度高，各项支持资源充沛。但容器技术与 PaaS 平台在国内市场还属于新兴技术范畴，与行业对接方面还有待验证。

2.2 项目意义

为推动新兴技术在银行业的应用，应对互联网金融服务对我行传统 IT 系统带来的技术冲击，中国光大银行积极探索容器与分布式微服务相关技术，推进开展基于开源技术的容器云 PaaS 平台建设项目。容器技术能大大提高系统应对大流量冲击时的并发处理能力，大幅缩短应用交付周期，提高部署配置效率和自动化运维水平。分布式微服务技术能有效沉淀行内公共数据与服务，快速

响应业务需求变化，提升应用服务数据化治理能力。秉承“统一规划，架构先行，技术创新，管理配套”的方针，通过对容器前沿技术和分布式微服务技术的研究和实践，充分发挥容器技术、分布式微服务技术与云计算技术融合的技术优势，建设自主、安全可控，具备持续交付能力、高资源利用率、连续高可用、灵活伸缩的自动化应用服务框架和 IT 管理体系架构。

该项目综合运用了业内领先的分布式微服务技术、容器技术、虚拟化&容器统一管理技术、资源及容器应用编排调度技术，软件负载均衡技术，软件定义网络及网络容器穿透技术等，采用循序渐进的方式，逐步对行内应用进行容器化改造，尝试进行应用的微服务架构转型，将单体应用逐渐进行微服务化拆分，同时探索和制定应用的模块化标准，提高相应的标准化程度。容器技术与分布式微服务技术的应用将大幅提高我行应用服务和运维的标准化和自动化水平，相应的管理和流程制度也需要逐步配合进行跟进与完善。尤其在该项目中进行分布式微服务、容器和云计算技术的有效整合，实现资源与应用的统一编排和调度，充分发挥分布式微服务、容器和云计算技术各自的优势，这在银行业采用分布式微服务和容器技术建设容器云 PaaS 平台，实现 IT 资源效能最大化和自动化运维方面具有十分重要的示范意义。

3 项目目标和范围

3.1 项目目标

3.1.1 总体目标

- 容器云 PaaS 平台建设项目预研阶段

1. 本项目具有技术创新和技术预研的重要特征，因此本期项目在项目目标设定上，同时具备预研项目的性质，也具备实际建设的具体目标。
2. 随着互联网技术的发展，庞大的单体应用形式对支持互联网应用越来越捉襟见肘。互联网应用具有如下特点：需求的持续发展和快速更新迭代、海量和波动的用户访问量，24 小时不间断的在线服务。这些技术特征要求应用可进行敏捷开发和快速迭代，支持海量并发和弹性伸缩，支持灰度发布和在线升级。分布式微服务和 PaaS 应用平台应运而生。分布式微服务将庞大的单体应用微服务化，轻量级服务带来的灵活性，可充分应对业务需求的快速变化，为我行业务的创新与快速发展，提供了有效的技术保障。而 PaaS 应用平台，可有效支撑微服务化拆分带来的开发测试部署运维复杂度提升，整合原来分布在各个单体应用中的重复功能和冗余数据，以共享服务的形式，沉淀我行的数据和服务能力。
3. 凭借类似集装箱的带来的高效，容器技术正在快速改变着公司和用户创建、发布、运行分布式应用的方式。根据市场研究公司 451 Research 的统计数据，2016 年容器技术市场的收入达到 7.62 亿美元（52.47 亿元人民币）。451 Research 还预测容器市场规模将于 2020 年接近 27 亿美元（185 亿元人民币），年复合增长率为 40%，发展比

OpenStack 更快。

4. 在技术层面容器平台和 IaaS 私有云平台可以做到无缝结合，在业务层面容器平台与 IaaS 私有云平台优势互补，容器云 PaaS 平台可运行在 IaaS 私有云平台之上，更加靠近应用层，更容易实现应用快速构建、持续集成、持续交付、应用快速发布、动态伸缩等场景。容器技术是 PaaS 的衍生，作为标准化的容器引擎优化 PaaS 的层级并拓展其应用范围。另一方面，容器作为自动化构建、发布的新模式，以轻量化、高效、可移植性等优势，释放容器和 IaaS 私有云 1+1 大于 2 的技术红利，为加快我行金融服务创新步伐，提高金融服务创新质量打下坚实的技术基础。

- 容器云 PaaS 平台建设

1. 容器云 PaaS 平台的架构应简单、高度集成，而不是过于复杂的架构设计，要简单且易维护。
2. 搭建分布式微服务框架，提供微服务的运行环境与数据化治理能力。
3. 为分布式微服务提供一系列分布式中间件服务与集成能力。
4. 解决开发、测试环境因为环境之间的差异导致部署效率低下、解耦多个项目共用同一个开发、测试环境的现状，构建 CI/CD 流水线，与代码仓库和 QA 等系统集成，构建从开发到测试的整个工作流。可持续测试的独立运行环境，便捷的测试数据回滚，及应用标准化部署。

5. 服务目录中的各项资源容器化，开发和测试团队协助更加高效、便捷的获取到所需的环境，开发和测试环境开箱即用。

- 利用容器云 PaaS 平台建设，本项目最终要实现的目标

1. 搭建运维层面的容器平台，提升服务器资源使用率，提供智能运维。
2. 搭建应用层面的 PaaS 应用平台，提供分布式微服务治理框架、云中间件服务与集成能力。
3. 建设应用开发流水线，提供微服务研发能力，打通开发测试运维 CI/CD 全流程，支持敏捷开发和快速业务响应。
4. 进行试点项目支持，并建立标准的开发规范和指南。

3.1.2 阶段目标

1、搭建应用层面的 PaaS 应用平台，提供分布式微服务治理框架、云中间件服务与集成能力。

✓ 搭建分布式微服务运行与治理框架：为分布式微服务提供高性能的运行平台、服务治理与数据化运维能力。提供服务注册、服务发现、服务路由、服务调用、服务监控、服务安全、故障恢复、轻量级通讯、链路跟踪、服务限流、服务降级、依赖管理、容量管理、应用生命周期管理、灰度发布、弹性伸缩、日志监控报警、故障隔离等诸多功能。

✓ 提供云中间件服务与集成能力：为分布式微服务提供负载均衡、API 网关、分布式缓存、分布式消息、分布式事务、分布式配置管理、分布式数据库等云

中间件服务与集成能力，支撑微服务在分布式环境下的运行。

2、搭建运维层面的容器平台，提升服务器资源使用率，提供智能运维。

✓ 提高配置部署效率和资源使用效率：容器能够根据需求量身定制，不必重复构建操作系统环境，提高基础环境配置部署效率并且最大化地利用服务器资源，保证安全的同时提高资源使用率，从而节约成本。

✓ 实现容器和虚拟化负载的统一管理：容器云 PaaS 平台统一管理容器和虚拟化负载，并且通过 SDN 容器穿透等技术实现在统一基础设施上运行不同类型负载的能力。

✓ 实现基于应用架构的资源交付：容器云 PaaS 平台实现了以应用架构为单位的资源交付模式。用户在云门户中可以选择各类标准化的应用架构，提升了用户资源申请效率。平台将行内各项技术规范和标准以自动化的方式进行固化和部署，确保了用户所得到资源的高度标准化，并符合业内最佳实践。

✓ 实现智能运维:容器云 PaaS 平台实现了资源和应用的统一编排和和统一监控，结合软件负载均衡技术可以实现资源和应用的自动化部署、弹性伸缩和自动故障隔离，确保应用的连续性和高可用性。

✓ 统一管理应用配置文件:容器云 PaaS 平台将统一管理各种应用的配置文件，容器云 PaaS 平台的管理配置文件功能，类似代码仓库版本控制，也可以实现配置文件每次的变更修改均保留新的版本，方便应用自动化管理（配置文件批

量下发、版本回退等），有效保证了应用配置文件的一致性。满足应用在多环境之间快速交付的需求，通过编排可设定默认值，在部署到我行开发、测试环境时结合具体应用的实际参数可进行修改，从而更好的实现了应用在多个环境之间的快速交付。可满足复杂应用集群的部署，配置文件可根据实际传入的参数进行动态更新，加速简化了复杂系统的部署和多部门间的交付。

3、建设应用开发流水线，提供微服务研发能力，打通开发测试运维 CI/CD 全流程，支持敏捷开发和快速业务响应。

✓ 通过与我行自主研发平台的集成，提供支持分布式微服务研发能力的微服务开发工具。对接我行云效平台，提供 CI/CD 能力，实现开发、构建、部署、测试、发布一体化的应用开发流水线。通过容器和镜像技术，容器云 PaaS 平台将应用运行环境包含在容器镜像中，这种具有可移植性的镜像可避免开发、测试、生产环境不一致导致的各种问题，提升了应用交付件的标准化，提高交付质量、降低风险。同时，通过打通开发到交付全流程，增进自动化水平，可有效支持敏捷开发，缩短交付周期，支持业务需求的快速响应。

4、进行试点项目支持，并输出一套基于容器云 PaaS 平台的分布式应用开发规范和指南

✓ 选择合适的试点项目，对试点应用的微服务化和容器化进行指导，对其使用应用开发流水线和在容器云 PaaS 平台上落地进行支持。

✓ 对于我行分布式应用的开发，要形成一套基于容器技术行之有效的开发规范及开发指南，按照规范和指南来开发和部署我行相对应的基于容器的分布式应用，从而为敏捷开发提供一套理论依据和可实行执行的方法。

3.2 项目范围

3.2.1 项目涉及范围

本系统的业务主管部门为总行信息科技部；业务需求提出方为总行信息科技部运维中心、开发中心、质量中心；总行信息科技部开发中心、质量中心、运维中心参与系统开发、性能测试和维护相关工作。

3.2.2 项目使用范围

本项目一期在光大银行总行推行，为总行搭建容器云 PaaS 平台，提供分布式微服务能力。

3.3 项目推广计划

一期：搭建容器云 PaaS 平台，提供分布式微服务能力。

二期：试点应用支持，不断完善平台功能。

4 项目建设需求

4.1 项目建设核心内容说明

4.1.1 基于 Docker 技术的容器平台

Docker 基于 Linux 内核的 Namespace、Cgroup 和 UnionFS（联合文件系统）等技术实现了环境的整体打包和标准化交付的。

容器平台以容器为资源分割和调度的基本单位，封装整个软件运行时环境，为开发者和系统管理员提供用于构建、发布和运行分布式应用的平台。利用 Docker、Kubernates 等技术，容器平台可大幅提高服务器资源使用率，并提供自动化和智能运维能力。

4.1.2 基于分布式微服务的 PaaS 应用平台

PaaS 应用平台是容器云在应用层级的实现。为分布式微服务提供运行环境与管理监控能力。对上通过网关为外部应用开放各种 API，对内支撑微服务群的生态环境，对下，通过调用容器平台接口，打通服务与资源的关联，达到服务+资源的弹性伸缩与自动化运维。

4.1.3 应用开发流水线

应用开发流水线，依托容器云 PaaS 平台，并对接我行云效平台，通过容器、镜像等技术，提供 CI/CD 能力。打通开发到交付全流程，实现编码开发、构建、部署、测试、发布一体化的应用开发流水线，增进自动化水平，有效支持敏捷开发，缩短交付周期，支持业务需求的快速响应。

4.1.4 试点项目支持

选择合适的试点项目，对试点应用的微服务化和容器化进行指导，对其使用应用开发流水线和在容器云 PaaS 平台上落地进行支持。

4.2 涉及工作流程说明

分类	测试场景	场景描述	场景类别
基础功能测试	用户管理	用户及用户组的添加、删除和资源授权等操作	功能
	平台总览	容器、主机、镜像、应用等总览信息	功能
	镜像管理	镜像的制作、推送、拉取等相关操作	功能
	网络管理	支持 SDN 网络穿透，网络性能损耗在百分之五以下	功能
	安全管理	操作审计、用户权限和资源隔离	功能
	配置文件管理	应用配置文件的快速更新和一致性	功能
	API 开放	API 对其他应用的开放	功能
	CI/CD 集成	Jenkins 的集成、GIT/SVN 代码库的对接等	功能
	应用发布	应用快速发布、版本管理，包含灰度发布，应用回滚	功能
	负载均衡	对接 nginx、F5 等负载均衡器	功能
	监控告警	服务故障邮件、短信和电话告警	功能
	服务目录	应用模板化，团队之间可以快速分享	功能
	应用管理	在线增加应用节点，支持应用伸缩，可提供定时伸缩策略和根据应用指标伸缩策略。	功能
	应用监控	在线查看应用健康状况，资源使用情况等监控视图。提供日志收集、查看与分析能力。	功能
	故障恢复	故障报告，定位与自动化故障恢复	功能
	服务治理	服务访问控制、服务跟踪、服务统计分析、流量控制、服务并发线程控制、多维度服务限流	功能
	分布式应用组件	分布式缓存/数据库/消息服务管理和高可用、对象存储服务管理和大文件上传支持、分布式事务处理机制、软负载均衡服务管理、API 网关服务管理	功能
	应用开发流水线	集成行内自动化集成测试平台，提供微服务研发能力和 devOps 能力。	功能

4.3 非功能性需求

考虑到本期项目为基础设施架构技术预研项目性质，所以性能需求是这次研究的重点，我们也将设计各种测试模型，用以考量以 Docker 技术作为容器云的整体性能表现以及针对我行实际业务的承载能力。包括不局限于以下需求：

- 易用性
- 可用性

- 可扩展性

4.3.1 性能

网络和容器负载需达到传统物理环境 90%的性能指标。

4.3.2 数据量预估

本期项目为技术预研项目性质，本期项目建设不以具体的平台或系统的投产为建设目标，因此未涉及生产数据的相关数据需求，考虑到容器基础镜像管理和自身元数据管理所需存储需求预估：元数据多副本在线保存约 1TB；基础镜像结合行内常用操作系统约 500GB，并通过数据副本方式满足数据备份、灾备恢复的管理要求。

4.3.3 可用性

- 容器平台系统可用率：　　不低于 99.9%
- 容器平台系统服务时间：　　7×24 小时

4.3.4 其他非功能性需求

安全性需求：隔离性与访问控制、容错与灾备

分类	测试场景	场景描述
安全性	用户资源隔离	验证不同用户间的资源隔离功能
	防火墙控制	验证防火墙对流量的控制
	网页安全	使用 appscan 对页面进行漏洞扫描

兼容性需求：主流虚拟化、操作系统

分类	测试场景	场景描述

兼容性	Windows	验证对 Windows 2008 操作系统的支持
	Redhat	验证对 Redhat 操作系统的支持
	Centos	验证对 Centos 操作系统的支持
	Ubuntu	验证对 Ubuntu 操作系统的支持

4.4 其他需要补充的内容

与云管理平台对接需求：北向接口完备程度与调用的难易程度

基础性能需求：基准测试、IO 延迟、长时间稳定性、极限性能、扩展性能

场景性能需求：我行 Oracle 典型交易压测—联机、批量

5 项目成本收益分析

5.1 项目成本

成本类型	项目成本(万元)			备注
	本年	后一年	后两年	
硬件采购成本	**0**	**0**	**0**	部署于私有云
软件采购成本	**200**	**0**	**0**	
项目实施成本	**234.4**	**220.8**	**196.8**	
项目成本合计	**434.4**	**220.8**	**196.8**	

5.2 项目收益

5.2.1 定量分析

收益类型	项目收益(万元)			备注
	本年	后一年	后两年	
收入类收益	**0**	**0**		
成本类收益	**0**	**240**	**480**	应用开发流水线提升效率减少工作量
风险类收益	**0**	**0**		
IT 类收益	**0**	**1080**		IT 人力资源成本减少及硬件复用效率提升
项目收益合计	**0**	**1320**	**480**	

注：

收入类收益：指项目建设所支撑的银行产品销售给客户后带来的收益；

成本类收益：指项目建设对加强管理和提升运营以及渠道完善带来的收益；

风险类收益：指项目建设对于加强和控制风险带来的收益；

IT 类收益：指项目建设使原材料成本和 IT 人力资源成本减少带来的收益。

5.2.2 定性分析

容器相较于虚拟机能够做到 1:32 的资源利用率，也就是说资源利用率是虚拟机的 4 倍，我行目前有 1200 个生产虚拟机（目前私有云单虚拟机成本约 2 万元），按照业界普遍 30%业务可以容器化标准计算，使用容器技术就能节省 1080 万元。

项目预计 2018 年 3 月份上线，同时基于此平台上线推广 1 个项目组，可以让应用项目组在开发，测试，部署、维护等环节节约 4 人左右。2018 年上线后，除试点项目外预计再推广 1 个项目，2019 年预计推广 4 个项目。

5.3 项目成本收益分析

财务数据	本年	后一年	后两年	总计
总收益(万元)	0	1320	480	1800
总成本(万元)	482.4	292.8	268.8	1044
净收益(万元)	(482.4)	1027.2	211.2	756
贴现率	8.00%			
现金流贴现(万元)	(446.6667)	880.6584	167.6574	601.6491
净现值(万元)				601.6491
效益成本比率				1.66

6 项目技术方案

6.1 技术方案设计原则

该项目综合运用了业内领先的分布式微服务技术、容器技术、虚拟化和容器统一管理技术、资源及容器应用编排调度技术，软件负载均衡技术，软件定义网络及网络容器穿透技术等，主要对传统应用架构中数据库前端的应用进行容器化。采用循序渐进的方式，逐步对行内应用进行容器化改造，尝试进行应用的微服务架构转型，将单体应用逐渐进行微服务化拆分，同时探索和制定应用的模块化标准，提高相应的标准化程度。

秉承“统一规划，架构先行，技术创新，管理配套”的方针，通过对容器前沿技术的研究和实践，充分发挥容器技术与云计算技术融合的技术优势，建设自主、安全可控，具备持续交付能力、高资源利用率、连续高可用、灵活伸缩的自动化应用服务框架和 IT 管理体系架构。

本项目设计原则：

- 成熟性

  使用业界成熟的 Docker 容器作为底层基本单元，国内大多数企业的容器云均为 Dcoker 架构，Docker 具有较强的公有云、私有云、混合云的扩展能力，为今后其它复杂、异构系统的融入和平台即服务解决方案的扩展提供支持。

- 开放性

  使用业界成熟的开源方案，向上层提供完善易用的 API 接口，对主机、

网络、负载均衡器等功能提供百分百 API 级别的支持，兼容主流操作系统。

- 扩展性

  项目着眼于基础架构云服务能力、计算能力、网络支持能力具备在线横向扩展特性。支持私有云、公有云和混合云部署，支持秒级扩容。

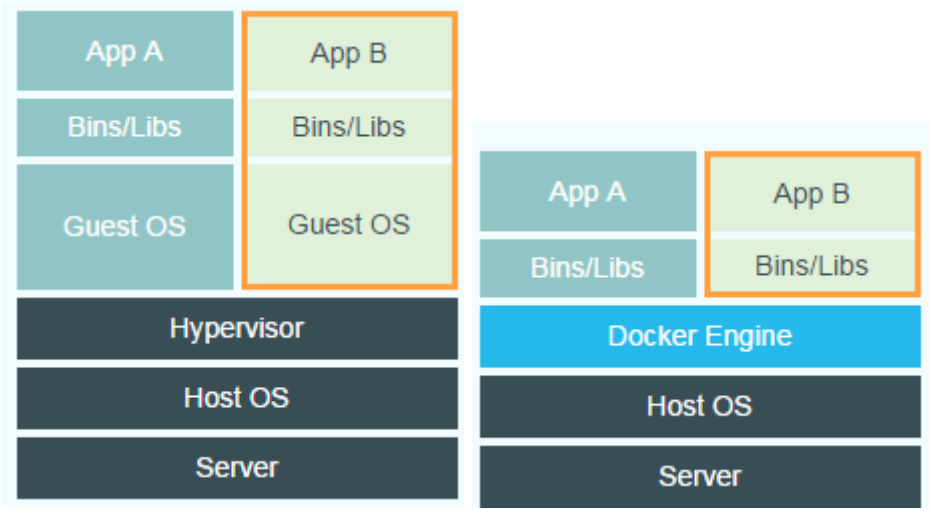
- 易用性

  提供管理和监控统一平台，提供基于 Web 的户自服务门户，具备成熟的自动化运维功能，具有简单易用的 Web 模拟终端支持，便于系统维护。系统自带资源实时监控和告警体系。

6.2 技术方案可行性分析

6.2.1 可行的技术方案比较

Docker 是近年来新兴的虚拟化工具，它可以和虚拟机一样实现资源和系统环境的隔离。下图为 Docker 与传统虚拟化原理的对比。



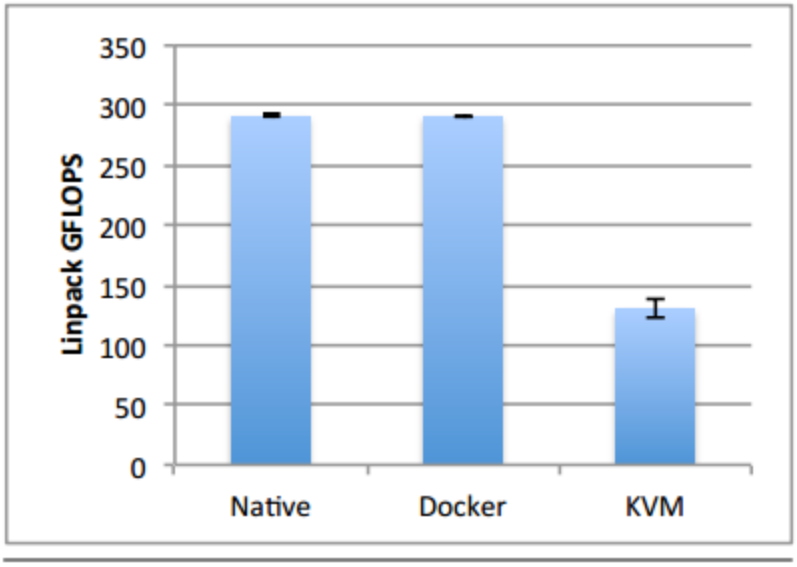
传统虚拟机 *Docker*

左图虚拟机的 Guest OS 层和 Hypervisor 层在 Docker 中被 Docker Engine 层所替代。虚拟机的 Guest OS 即为虚拟机安装的操作系统，它是一个完整操作系统内核；虚拟机的 Hypervisor 层可以简单理解为一个硬件虚拟化平台，它在 Host OS 是以内核态的驱动存在的。可以理解为如下：

- Docker 有着比虚拟机更少的抽象层。由于 Docker 不需要 Hypervisor 实现硬件资源虚拟化，运行在 Docker 容器上的程序直接使用的都是实际物理机的硬件资源。因此在 CPU、内存利用率上 Docker 将会在效率上有优势，在 IO 设备虚拟化上，Docker 的镜像管理有多种方案，比如利用 Aufs 文件系统或者 Device Mapper 实现 Docker 的文件管理，各种实现方案的效率略有不同。
- Docker 利用的是宿主机的内核，而不需要 Guest OS。因此，当新建一个容器时，Docker 不需要和虚拟机一样重新加载一个操作系统内核。引导、加载操作系统内核是一个比较费时费资源的过程，当新建一个虚拟机时，虚拟机软件需要加载 Guest OS，这个新建过程是分钟级别的而 Docker 由于直接利用宿主机的操作系统，则省略了这个过程，因此新建一个 Docker 容器只需要几秒钟。另外，现代操作系统是复杂的系统，在一台物理机上新增加一个操作系统的资源开销是比较大的，因此，Docker 对比虚拟机在资源消耗上也占有比较大的优势。事实上，在一

台物理机上我们可以很容易建立成百上千的容器，而只能建立几个虚拟机。

- 根据 IBM 发表的论文中的数据显示，Docker 相对于物理机其计算能力几乎没有损耗，而虚拟机对比物理机则有着非常明显的损耗。虚拟机的计算能力损耗在 50%左右。以下数据数据是在 IBM x3650 M4 服务器测得，其主要的硬件参数是：2 颗英特尔 xeon E5-2655 处理器，主频 2.4-3.0 GHz。每颗处理器有 8 个核，因此总共有 16 个核，256 GB RAM，在测试中是通过运算 Linpack 程序来获得计算能力数据的。



图中从左往右分别是物理机、Docker 和虚拟机的计算能力数据

两种技术的横向对比

对比项	传统虚拟机	Docker
调试镜像成本	高	低
占用磁盘空间	高	低

镜像大小	GB	MB
交付难度	较难	易
软件包(rpm/deb)	复杂的有向依赖	单继承、UnionFS
可伸缩性	低	高
运营成本	较高	低

6.2.2 新技术引入说明

本期项目具有技术创新和技术预研的重要特征，为技术预研项目性质，本期项目容器云 PaaS 平台研发目标和范围涉及较多新技术的使用，这些技术通过预研验证后，可能会产生相应的新技术引入需求，本项目可能涉及但不局限于以下技术框架：

- Docke 引擎
- Kubernates 服务编排调度引擎
- Dubbo/Spring Cloud 等分布式微服务技术框架

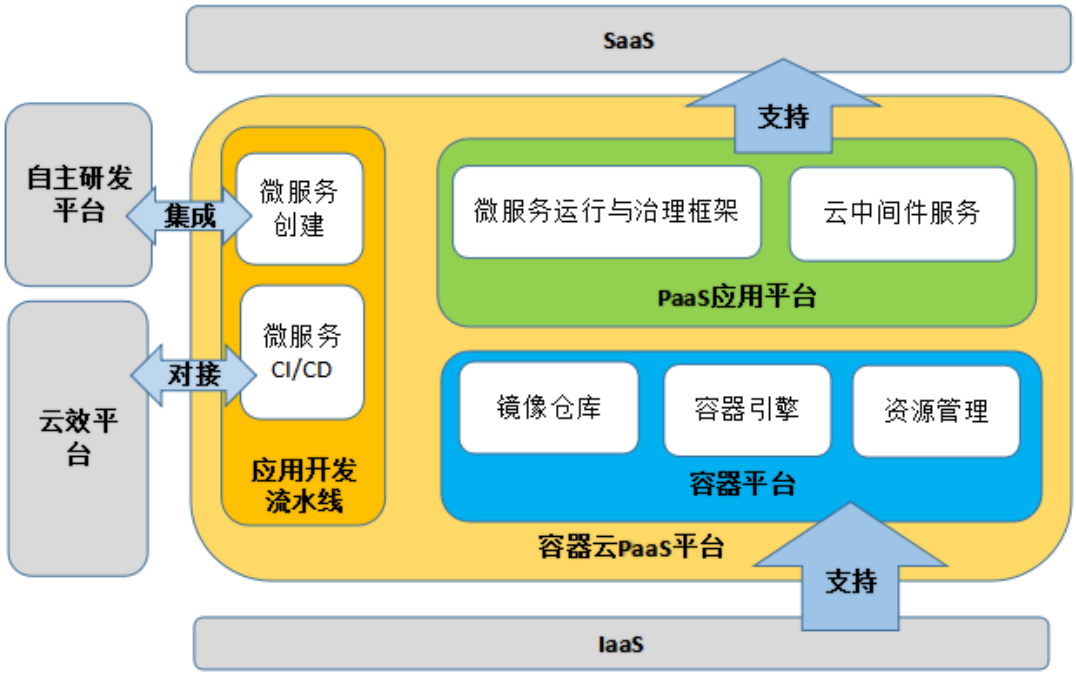
6.2.3 第三方产品引入说明

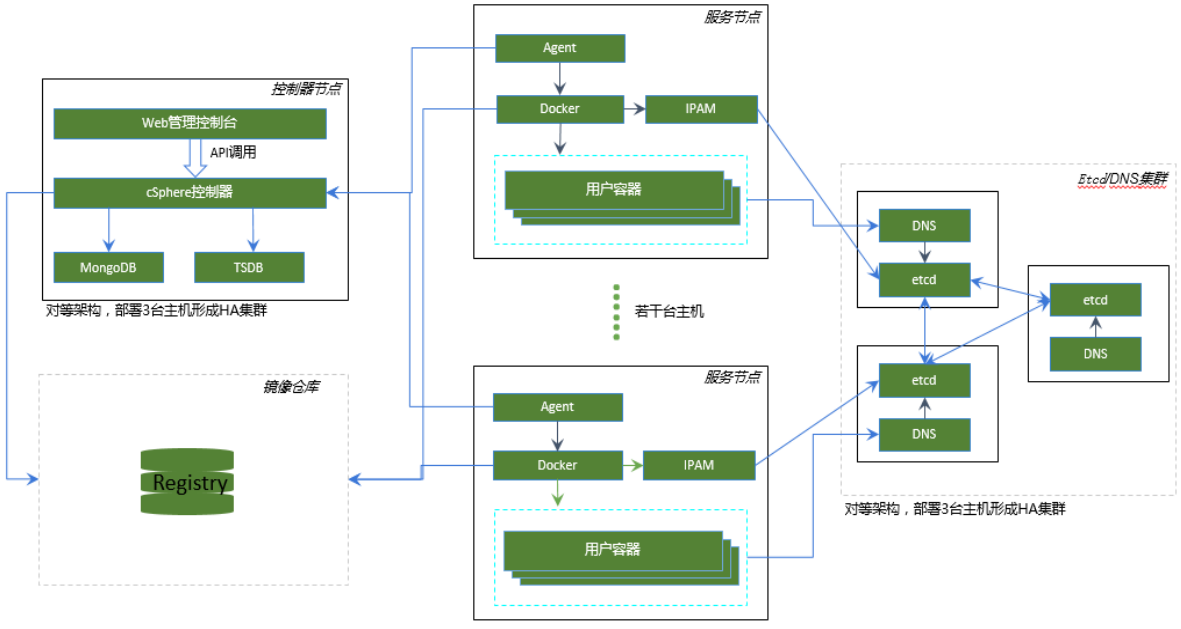
截止立项研究阶段，本期项目涉及的容器及分布式微服务研究均为市场主流的开源技术框架，未涉及到商业第三方产品的引入。

技术方案架构设计

6.3 技术方案架构设计

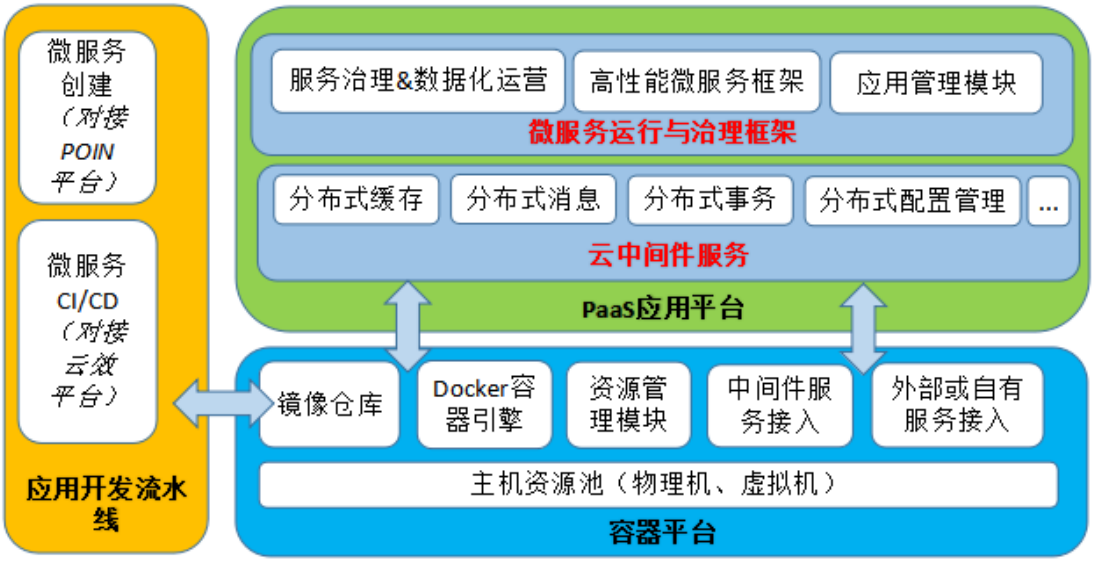
6.3.1 总体架构





6.3.2 逻辑架构

容器云 PaaS 平台在功能上分为三部分：容器平台、PaaS 应用平台、应用开发流水线。

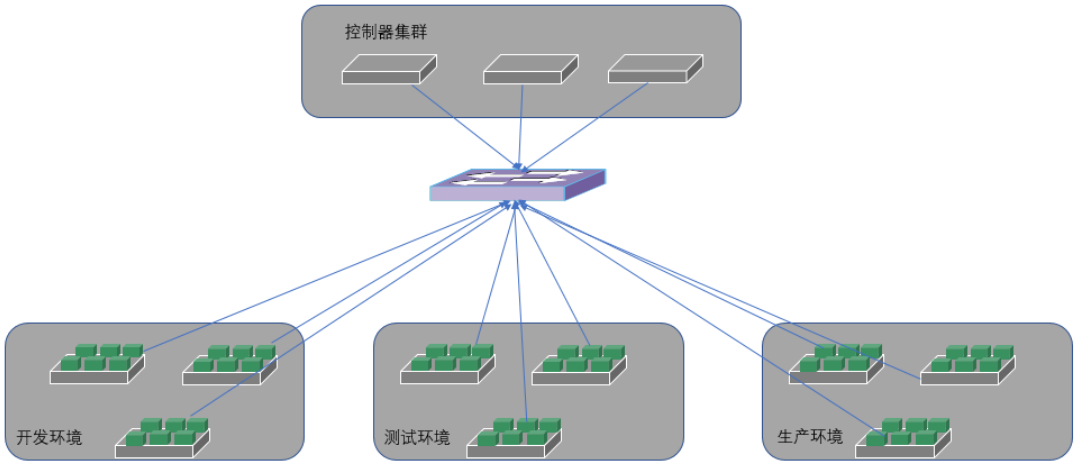


其中容器平台详细功能如下图：

1. 主机资源池为物理机或虚拟机，对容器提供基础的运行环境。

2. 容器引擎：Docker 引擎

6.3.3 物理架构



6.4 主要功能模块设计

6.4.1 容器平台

6.4.1.1 容器编排引擎模块

容器编排引擎是容器云的基础，通过编排形成应用模板，将应用运行环境作为代码一样来管理，达到编排一次，随时随地可以复用的效果。可以自定义资源配额、日志配置、服务启动优先级等。

6.4.1.2 容器调度引擎模块

当容器集群达到一定规模的时候，人工决定容器创建于哪些机器上回比较麻烦，且无法有效利用资源，容器调度引擎模块可以实现根据 CPU 利用率、内存剩余量、容器数量等策略来自主选择容器创建到哪台机器上。

6.4.1.3 容器部署引擎模块

应用生命周期是容器云的核心。通过容器部署引擎模块，可以快速的把应用

模板的服务各种服务部署到主机上，通过可视化的界面跟踪应用运行状态对应用进行各种管理，包括应用生命周期管理、容器生命周期管理以及容器配额管理。

6.4.1.4 容器配置引擎模块

无需重建容器和容器镜像，快速更新容器中的配置文件，并保证同一个服务分布于不同机器上的容器中配置的一致性。支持 Post Script，当容器配置文件分发完成后，自动执行自定义的脚本文件，进行诸如重新加载配置文件、容器重启等类型的工作，极大简化运维人员的工作流程。可以通过可编程的的模板语言动态创建配置文件，有极大的配置灵活性。还可以自动获取相关服务参数，诸如，Web 服务中的所有容器的 IP 列表，生成相关的配置文件。

6.4.1.5 容器健康引擎模块

容器的健康状态直接影响到 SLA 的水平，容器健康引擎模块可以自动探测容器的健康状态，并按需修复不健康的容器。该模块包括服务健康检查，如通过 TCP 端口检查连通性，自定义脚本检查。故障自动恢复，可以通过提前配置容器异常时的响应动作，如容器重启、容器重建等。所有服务的健康状态均可以通过可视化的状态实时查看。

6.4.1.6 容器构建引擎模块

可以和内部的 GIT、SVN 等代码管理系统进行无侵入性无缝整合，当开发者想代码库提交代码时，构建引擎可以自动从代码库获取代码并构建容器镜像，可按需自动上线服务，可以根据自己实际需求上架自己的构建包，快速进行构

建内部应用。

6.4.1.7 容器监控引擎模块

在容器云 PaaS 平台运行的过程中，管理员经常需要对整个容器云的资源消耗有全局的了解，当出现异常时，管理员和运维人员需要能够及时响应。容器监控引擎模块可以实时获取容器、主机的 CPU、内存、网络、磁盘读写速率、磁盘容量等各种资源的利用率，并可以配置灵活的告警规则，将其与被监控的资源进行绑定，支持 HTTP Hook 方式告警，实现实时发送告警短信、拨打告警电话等功能。以上监控各种监控数据实时显示在平台的仪表盘并动态更新。

6.4.1.8 容器服务发现引擎模块

在复杂的内部应用环境中，一个应用往往会涉及到多个相互依赖的服务。如 Web 服务需要连接数据库、缓存、消息中间件等服务；数据库从库需要连接主库同步数据；消息队列需要连接队列服务；在服务中存在依赖关系时，被依赖的 IP、端口、以及各种连接参数管理较为麻烦，尤其是在容器云 PaaS 平台上，若是没有服务发现机制处理这些连接参数，将无法实现服务的弹性伸缩。该模块基于 DNS 实现，该模块会给每个容器分配一个域名，并结合 DNS 内置的 search 选项提供短域名解析，应用代码中只要通过域名或者服务名字，即可与其依赖的的服务进行通信，从而保证同一套代码多个副本部署时无需调整任何配置即可直接运行。当服务动态伸缩时，DNS 记录会实时更新。保证服务中的容器发生变化时，应用能够实时感受到变化。

6.4.2 PaaS 应用平台

6.4.2.1 服务治理与数据化运营

服务治理与数据化运营，提供链路跟踪、服务限流、服务降级、依赖管理、调用分析、容量管理等功能。

6.4.2.2 高性能微服务框架

高性能微服务框架，支撑微服务群的运行，包含：服务注册、服务发现、服务路由、服务调用、服务监控、服务安全、故障恢复、轻量级通讯等功能。

6.4.2.3 应用管理模块

应用管理模块，提供应用接入、开通、使用、下线、灰度发布、应用级弹性伸缩、监控、故障隔离、报警通知、日志管理等功能。

6.4.2.4 云中间件服务与集成能力

云中间件服务提供分布式数据层、分布式缓存、分布式消息、分布式事务、分布式配置管理、分布式批量引擎、分布式 sequence、服务网关等服务能力与集成能力。

6.4.3 应用开发流水线

与容器云 PaaS 平台集成，对接云效平台，提供 CI/CD 能力，打通微服务创建到交付的全流程。支持敏捷开发，提供 DevOps 能力。

对已有应用提供容器化能力。

6.4.4 试点项目支持

6.4.4.1 试点项目支持

选择合适的试点项目，对试点应用的微服务化和容器化进行指导，对其使用应用开发流水线和在容器云 PaaS 平台上落地进行支持。

6.4.4.2 基于容器云 PaaS 平台的分布式应用开发规范和指南

对我行分布式应用的开发，形成一套行之有效的开发规范及开发指南，按照规范和指南来支持试点项目的开发和部署，从实践中总结沉淀经验并改进规范和指南，更好的支持我行基于容器云 PaaS 平台的分布式应用研发，从而为敏捷开发提供一套理论依据和可实行执行的方法。

6.5 与其他系统关系

平台搭建在我行 IaaS 虚拟云基础上，建成后可支持我行 SaaS 的建设。通过与我行自主研发平台集成和对接云效平台，打通开发到交付全流程。监控信息供给统一监控平台。

6.6 第三方系统接口

无

6.7 系统运行环境

6.7.1 硬件环境

编号	硬件类别	硬件用途	拟选用产品及配置要求	数量(台)
1	X86 服务器	部署容器控制节点	高性能 X86 服务器	4

2	X86 服务器	部署容器 Agent 节点	高性能 X86 服务器	4

6.7.2 软件环境

编号	软件类别	软件功能	拟选用产品及版本要求	数量(套)
1	操作系统	为容器提供基础运行环境	CentOS 7.2.1511	1
2	容器引擎	提供容器管理功能	Docker Engine 1.11.2	1

根据试点规模适当调整。

7 项目实施方案

7.1 项目实施安排

7.1.1 项目工期及里程碑

整个项目周期预计从 2017 年 06 月到 2018 年 3 月，为期 10 个月。

编号	时间	项目里程碑说明	备注
1	2017 年 06 月	完成项目立项、招标工作，商务谈判	
2	2017 年 09 月	完成需求说明书，需求分析，完成设计	
3	2017 年 12 月	完成开发和技术测试	
4	2018 年 02 月	完成系统测试、联调测试、性能调优	
5	2018 年 03 月	完成投产上线、项目文档整理、试点应用支持	

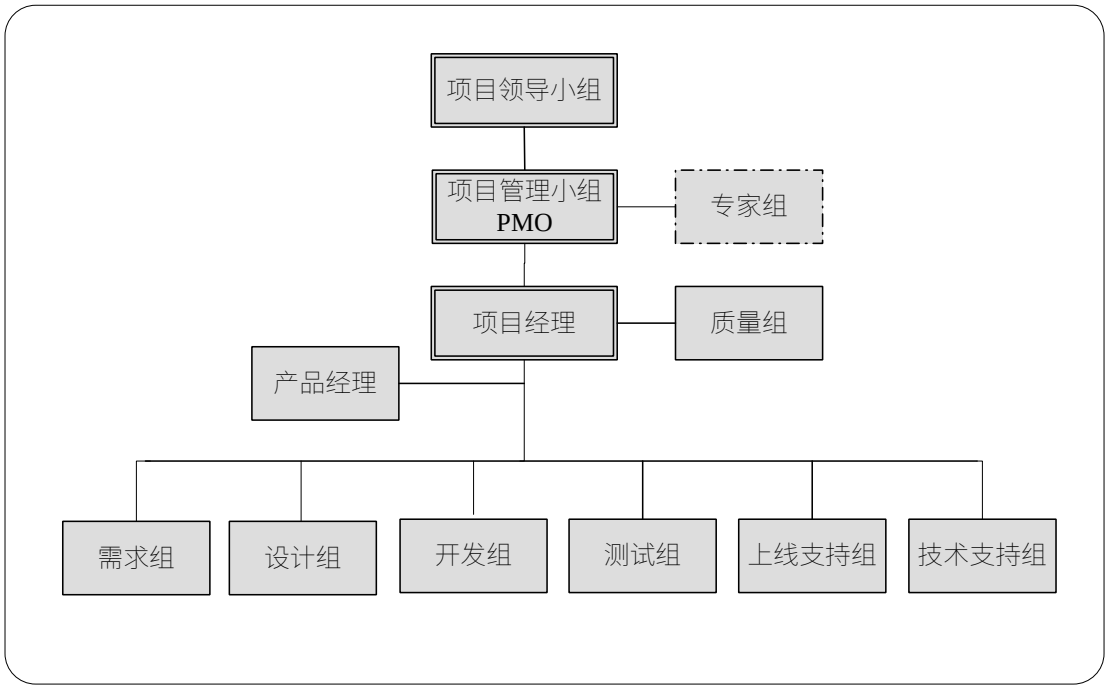
7.1.2 项目实施计划

编号	起止时间	项目任务说明	备注
1	2017 年 09 月- 2017 年 09 月	平台需求分析和系统设计	
2	2017 年 9 月–2018 年 12 月	平台开发阶段	
3	2017 年 12 月-2018 年 02 月	平台系统测试&联调测试&性能调优	
4	2018 年 03 月–2018 年 03	投产及推广	

	月		
5	2017 年 11 月–2018 年 03 月	试点应用支持和项目文档整理	

7.1.3 项目人力资源安排

项目组织结构：



项目人力资源安排：

编号	项目角色	人员数量(名)		备注
		行内人员	外包人员	
1	项目领导小组	2	0	
2	项目管理小组	2	0	
3	项目经理	2	1	
4	开发组	1	12	
5	技术支持组	1	1	
6	测试支持组	1	4	
7	质量管理组	1	1	

7.1.4 项目商务方式建议

公开招标

8 项目风险评估

8.1 项目主要风险因素分析

近年来，国家有关部门和监管机构日益重视银行业信息科技风险管理工作，要求商业银行加强信息科技风险管理，实现对信息系统的“安全、可控”。银行业要围绕“自主可控”、“持续发展”、“科技创新”三大战略切实加强信息科技建设，要求银行业金融机构按照“风险可控”、“自主可控”战略，合理构建信息系统。容器云 PaaS 平台建设项目是由科技部主导进行项目建设，无具体的业务部门，在项目风险方面，主要存在运营风险、数据安全风险、项目外包风险，具体风险与应对措施如下：

- 为控制系统投产后的运营风险，满足 IT 系统安全运营管理要求，保障生产系统安全稳定运行，在项目架构设计时充分考虑平台的安全稳定运行方案。在平台数据节点上采用多副本实时数据备份策略，在单个数据节点宕机时，可通过其他数据节点副本对外部业务应用提供服务，降低对应用的影响，保障平台的高可靠；平台服务器采用分布式架构，方便节点横向扩展，在任何一个节点资源不足时，可实时动态调整系统资源，保障服务器资源的高可用；平台对外服务的前端应用服务器上，均为负载均衡方式且冗余度高于 100%，可保证 7*24 小时对外提供服务；
- 容器云 PaaS 平台预计将承载着我行各类系统的业务数据，其中包括各种机密、隐私数据，为严格控制数据风险，保障数据安全可用，平台

采支持多种安全架构设计，多用户架构提供强大的隔离措施。通过物理机操作系统内核的资源隔离的算法保证每个业务应用系统只能访问自身权限范围内的数据及计算资源，任何数据访问与平台功能调用都将记录日志，可为后续审计时提供数据，保障平台内数据使用安全；

- 根据当前技术发展的现在，任何企业的容器云 PaaS 平台系统都不能在硬件、软件方面完全做到自主研发，必须要与厂商进行合作开发，在与外包厂商合作时，隐含存在项目外包风险。在选择合作厂商时，需要厂商严格遵守我行外包管理规定，签署合作保密协议，提供厂商基本信息、财务信息、产品信息、案例信息、项目组人员简历信息等，只有满足供应商评估要求后，方可进行项目合作。进入项目的开发人员必须参加外包人员准入考试，合格后才能入场，在项目组现场办公的人员，必须使用我行提供的外包专用机，防止内部数据外泄。

8.2 项目风险度计算结果

根据项目风险度计算模型，计算出项目风险度得分为 5.55。

条目	项目规模	项目复杂度	项目相关性	项目经验
项目数据	预算 434.4	118 人月	完全独立	新的开发工具类型
分值	5	6	1	8
项目风险度得分	项目规模分值*0.2+项目复杂度分值*0.4+项目相关性分值*0.15+项目经验分值*0.25 = 项目风险度得分 5.55			

8.3 安全风险度计算结果

根据安全风险度计算模型，计算出安全风险度得分为 1.7，项目安全风险度等级为低级。

条目	系统用户	与第三方的数据传输	系统安全保护等级
项目数据	科技部内部用户	无需第三方数据传输	等保一级
分值	2	1	2
安全风险度得分	系统用户分值*0.3+与第三方数据传输分值*0.3+系统安全保护等级分值*0.4 =安全风险度得分 1.7		

9 结论

基于以上项目意义可以看出，容器云 PaaS 平台项目建设，对于我行应用研发、应用交付、自动化运维和 PaaS 平台全面建设预研有着重要意义，并且对我行整体 IT 能力和水平的建设和提升有着促进作用，同时考虑到目前容器云 PaaS 平台建设紧迫程度，建议尽快展开工作。

附件

无