ICS 点击此处添加ICS号

点击此处添加中国标准文献分类号

|  |
| --- |
|  |

中华人民共和国     行业标准

XX/T XXXXX—XXXX

|  |
| --- |
|  |

研发运营一体化（DevOps）能力成熟度模型

第9部分 系统和工具

点击此处添加标准英文译名The Capability Maturity Model of DevOps

Part 9 ：System and Tools

点击此处添加与国际标准一致性程度的标识

|  |
| --- |
|  |
|  |

XXXX - XX - XX发布

XXXX - XX - XX实施

        发布

目录

[前言 II](#_Toc524536392)

[研发运营一体化（DevOps）能力成熟度模型 第9部分：系统和工具 1](#_Toc524536393)

[1 范围 1](#_Toc524536394)

[2 规范性引用文件 1](#_Toc524536395)

[3 术语 1](#_Toc524536396)

[3.1 配置项 configuration item（样例）](#_Toc524536397) **[错误!未定义书签。](#_Toc524536397)**

[3.2 制品 artifact（样例）](#_Toc524536398) **[错误!未定义书签。](#_Toc524536398)**

[3.3 部署流水线 deployment pipeline（样例）](#_Toc524536399) **[错误!未定义书签。](#_Toc524536399)**

[3.4 xxx （英文）](#_Toc524536400) **[错误!未定义书签。](#_Toc524536400)**

[4 缩略语 1](#_Toc524536401)

[5 总体架构 9](#_Toc524536402)

[5.1 项目与开发管理 9](#_Toc524536403)

[5.2 应用设计与开发 9](#_Toc524536404)

[5.3 持续交付 14](#_Toc524536405)

[5.4 测试管理 22](#_Toc524536406)

[5.5 自动化测试 24](#_Toc524536407)

[5.6 安全开发 29](#_Toc524536408)

[5.7 安全交付 29](#_Toc524536409)

[5.8 安全运营 30](#_Toc524536410)

前  言

研发运营一体化是指在IT软件及相关服务的研发及交付过程中，将应用的需求、开发、测试、部署和运营统一起来，基于整个组织的协作和应用架构的优化，实现敏捷开发、持续交付和应用运营的无缝集成。帮助企业提升IT效能，在保证稳定的同时，快速交付高质量的软件及服务，灵活应对快速变化的业务需求和市场环境。

本标准是“研发运营一体化（DevOps）能力成熟度模型”系列标准的第 9 部分 系统和工具，该系列标准的结构和名称如下:

第1部分：总体架构

第2部分：敏捷开发管理

第3部分：持续交付

第4部分：技术运营

第5部分：应用设计

第6部分：风险管理

第7部分：组织结构

第8部分：评估方法

第9部分：系统和工具

本标准按照GB/T 1.1-2009给出的规则起草。

本标准由中国通信标准化协会提出并归口。

本标准起草单位： 待完善

本标准主要起草人：待完善

研发运营一体化（DevOps）能力成熟度模型 第9部分：系统和工具

1. 范围

本标准规定了研发运营一体化（DevOps）的xxxxx。

本标准适用于具备IT软件研发交付运营能力的组织实施IT软件开发和服务过程的能力进行评价和指导；可供其他相关行业或组织进行参考；也可作为第三方权威评估机构衡量软件开发交付成熟的标准依据。

1. 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

[1] GB/T 32400-2015 信息技术 云计算 概览与词汇

[2] GB/T 32399-2016 信息技术 云计算 参考架构

[3] YD/2441-2013 互联网数据中心技术及分级分类标准

[4] GB/T 33136-2016 信息技术服务数据中心服务能力成熟度模型

[5] GB/T 31496-2015 信息技术 安全技术 信息安全管理体系实施指南

[6] GB/T 22080-2016 信息技术 安全技术 信息安全管理体系要求

[7] GB/T 30975-2014 信息技术 基于计算机的软件系统的性能测量与评级

1. 术语

下列术语和定义适用于本文件。

* 1. 项目 project

需要协同工作的一组任务，其目的在于开发和（或）维护一个具体的产品。产品可以包括硬件、软件或其他成分，一般项目有自己的经费、成本核算和交付进度。【GB/T11457-2006：2.954】

* 1. 【董越】项目集 program

PMI将项目集定义为经过协调管理以获取单独管理所无法取得的收益的一组项目、子项目集和项目集活动。

* 1. 【董越】子项目集 sub program

作为另一个项目集的组成部分而被管理的一个项目集

* 1. 构建项目  Build project

作为一个构建软件的配置，以及流程。

* 1. 执行记录  Build history

执行构建的信息，以及结果。

* 1. 构建产出物  Build output

构建项目中生产的文件。

* 1. 凭据 Credentials

保存用户保密信息，例如：用户名和密码。

* 1. 容器镜像 Container Image

容器技术中运行容器软件。包含基本操作系统以及依赖的软件。可以包含构建的产出物。

* 1. 运行环境 Runtime environment

构建中运行的环境或者计算机。

* 1. 源代码托管软件 Source code management software

保存源代码的软件。

* 1. 软件单元测试 Software unit test

软件中研发的自动化测试。

* 1. 发布

发布通常是软件程序的变动，将线上的流量转移到新的软件版本的过程

* 1. 原地发布

将新的软件版本直接推送到服务器上覆盖旧版本，发布过程可能会引起服务的中断，是一种最直接的发布方式

* 1. 金丝雀发布

是指在黑与白之间，能够平滑过渡的一种发布方式。又称灰度发布，A/B测试就是一种灰度发布方式，让一部分用户继续用A，一部分用户开始用B，如果用户对B没有什么反对意见，那么逐步扩大范围，把所有用户都迁移到B上面来。可以保证整体系统的稳定，在初始灰度的时候就可以发现、调整问题，以保证其影响度范围的一种发布方式

* 1. 蓝绿发布

在发布过程中，新旧版本相互热备，发布过程是不停止老版本，直接部署新版本然后进行测试，通过切换路由权重的方式（非0即100）实现将流量切到新版本，然后老版本同时也升级到新版本，是一种可以保证系统在不间断提供服务的情况下的发布方式

* 1. 回滚

回滚（Rollback）指的是程序或数据处理错误，将程序或数据恢复到上一次正确状态的行为。回滚包括程序回滚和数据回滚等类型

* 1. 环境 Environment

是指应用程序运行所需的所有资源和它们的配置信息，通常分成两大类：基础设施资源及其配置；应用程序独立运行所需要的操作系统和中间件资源及其配置信息。

* 1. 配置项 Configuration item

一个配置中的实体，它满足一项最终使用功能，并能在给定的基准点上单独标识。【GB/T 8566-2001：3.6】

* 1. 配置管理 Configuration Management

~~是一个过程，通过该过程，可以维护一切与环境相关的信息，这些信息可以被定义、修改、存储和检索。~~应用技术和管理的指导和监控方法以标识和说明配置项的功能和物理特征，控制这些特征的变更，记录和报告变更处理和实现状态并验证与规定的需求的遵循性。【GB/T11457-2006：2.313】

* 1. 开发环境 Development Environment

用于维持开发过程的一个或者一组资源及其配置信息，面向的是开发者。

* 1. 测试环境 Test Environment

用于维持测试过程的一个或者一组资源及其配置信息，面向的是测试者。

* 1. 生产环境 Production Environment

用于维持开发过程的一个或者一组资源及其配置信息，面向的是运维，但其运行服务面向的是用户或者客户。

* 1. 基础设施 Infrastructure

支持应用运行的所有服务，包括DNS服务器、防火墙、路由器等等。

* 1. 中间件 Middleware

~~是处于操作系统和应用程序之间的软件。~~一种类型的软件模块，它处在系统软件和应用软件之间，依赖系统软件的支持，又为应用软件提供支持，以方便应用软件的开发。【GB/T11457-2006：2.954】

* 1. 性能测试（performance test）

~~性能测试是通过自动化的测试工具模拟多种正常、峰值以及异常负载条件来对系统的各项性能指标进行测试。~~评价系统或部件与规定的性能需求的依从性的测试行为。【GB/T11457-2006：2.1135】

* 1. 负载测试（load testing）

通过测试系统在资源超负荷情况下的表现，以发现设计上的错误或验证系统的负载能力。在这种测试中，将使测试对象承担不同的工作量，以评测和评估测试对象在不同工作量条件下的性能行为，以及持续正常运行的能力。

* 1. 压力测试（stress testing）

压力测试是模拟实际应用的软硬件环境及用户使用过程的系统负荷，长时间或超大负荷地运行测试软件，来测试被测系统的性能、可靠性、稳定性等。

* 1. 稳定性测试（stability test）

通常是采用系统稳定运行情况下能够支持的最大并发用户数或者日常运行用户数，持续执行一段时间业务，通过综合分析交易执行指标和资源监控指标来确定系统处理最大工作量强度性能的过程。

* 1. 活动（activity）

由用户或模拟用户向在册系统（SUT）提交的一种命令：要求依界定的算法来执行数据处理操作，从特定输入数据和（若要求时的）既存数据产生特定输出数据。

* 1. 活动类型（activity type）

对执行同一算法所界定的各种活动的分类。

* 1. 链（chain）

以界定的顺序向STU提交的一个或不止一个任务。

* 1. 链型（chain type）

对由任务（类）型序列所界定的链所作的分类。

模拟用户只向SUT提交规定链型的链。

所指链型当前序号的数学符号为L，链型总数的数学符号为u。

* 1. 模拟用户（emulated user）

就用户提交任务及其时变行为实施模仿的技术系统。

* 1. 执行时间（execution time）

从任务提交至任务完成所经历的时间。

* 1. 平均执行时间（mean execution time）

在评级区间内提交的第j型所有任务的执行时间的均值。

* 1. 平均执行时间参考值（mean execution time reference value）

模拟用户可接受的最大平均执行时间。

* 1. 响应时间（response time）

指用户发起一个请求开始，服务器完成对该请求的处理并返回处理结果所经过的时间。

* 1. 观测期（observation period）

为汇集（日志记载）用户评级或确认的测量结果而观测测量规程的时间区间。由评级区间和补充运行时间组成。

* 1. 准备时间(perparation time)

任务提交之前经历的时间。启动准备时间的事件取决于对后一任务的任务模式的界定。准备时间的值是带有界定的均值和标准差的分布式变量的随机值，这取决于后一任务的任务类型和生成此任务的模拟用户的类型。

* 1. 评级区间(rating interval)

测量规定从SUT达到操作稳定状态的时间开始，到测量结果满足所需统计显著性的时间为止的时间区间。

* 1. 相对链频(relative chain frequency)

第i型的模拟用户使用第l型链的相对频数。

* 1. 稳定化阶段（stabilization phase）

测量规程从RTE提交任务开始，到SUT达到稳定操作状态位置的时间区间。

* 1. 补充运行时间（supplementary run）

测量规程从测量结果达到满足所需统计显著性的时间开始，到评级期间已提交的所有任务全部完成时间为止的时间区间。

* 1. 在测系统（system under test，SUT）

CBSS中接受测试的个部分。影响SUT时变行为的所有部件都应是SUT的组成部分，如果此种影响取决于某一工作负载，则该负载也应由RTE来标识。除非对时变行为的影响不可能或不明显时，对其余部分的模拟才可省略。

* 1. 任务（task）

特定活动，由特定适时性函数界定的所需执行时间，特定任务模式。

* 1. 任务完成（task completion）

对特定任务的如下适时时间：由模拟用户或另一实例全部收到整个输出串，或者在输出串集的情况，收到该集合的所有部分。该任务未向用户（即测量期间向RTE）提交输出时，可为该任务添加一个功能，即产生“人工输出”来指明该任务完成，供测量期间使用。

* 1. 任务模式(task mode)

对用户的准备时间是在前一任务提交后立即开始（值=0，即“不等待”）还是当前一任务完成时（任务完成）开始（值=1，即“等待”）所做的指示。

* 1. 任务提交（task submission）

当输入串完整的从模拟用户提交SUT时，该任务即可开始执行，而不管SUT是否立即执行的适时事件。任务提交还可通过这一任务的模拟用户行动结束时间来指示。根据本标准，任务提交不应该由以下事件来界定（和测量）：SUT收到或解释完输出串，或模拟用户接收一个可由SUT在接收和解释之后送出的收据串。

* 1. 任务类型（task type）

任务所有分类：

活动类型，或属于同一实时性函数和任务模式的所有活动类型的集合；

适时性函数；

任务模式。

* 1. 吞吐量（throughput）

提交给SUT某一任务型所有任务的速率（即评级区间内单位时间的平均数）

* 1. 吞吐量参考值（throughput reference value）

实际吞吐量与吞吐量参考值之比（对应于第j任务型）

* 1. 时间类别（time class）

与一种相对频数相结合，用于和（特指任务型的）任务的执行时间相比较的时限。其中的相对频数对应于执行时间小于等于对应时限的（特定任务型的）任务数和（该特指任务型）任务总数之比。

* 1. 实时吞吐量（timely throughput）

对应于适时性函数，执行时得以接收的所有任务的吞吐量。

* 1. 用户（user）

通过终端（或等效的机器用户接口）使用CBSS功能来提交任务并接收算出结果的人（或实例）。

* 1. 虚拟用户（virtual user）

通过函数或任务模拟的方式完成等效机器用户接口功能，使用CBSS功能来提交任务并行接收算出结果的特定程序集合。

* 1. 最大并发用户数（maximum number of concurrent users）

系统能够承受的同时使用系统服务或资源的用户数的极限，超过该用户数，将导致系统效率严重下滑，并可能导致系统崩溃、失效。

* 1. 最大并发请求数（maximum number of concurrent users）

系统能够承受的同时接收到的请求数的极限，超过该请求数，将导致系统效率严重下滑，并可能导致系统崩溃、失效。

* 1. 用户类型（user type）

对由下列两项的组合所界定的模拟用户的一种分类：

使用链类型的相对频数；

准备时间（均值及其标准差）

* 1. 事务（transaction）

事务是脚本的一个特性，每个事务都包含开始事务和结束事务。事务用来衡量脚本中一行代码或多行代码的执行所耗费的时间。你可以将开始事务放置在脚本中某行代码的前面，将结束事务放置在该行代码的后面，在该脚本的虚拟用户运行时，这个事务将衡量该行代码的执行花费了多长时间。

* 1. 事务吞吐容量（transaction throughput capacity）

在一定的时间单位内，系统能够处理完成的最大事务量。

* 1. 数据吞吐容量（data throughput capacity）

在一定的时间单位内，系统能够处理完成的最大数据量。

* 1. 延迟（delay）

完成一个请求的延时。

* 1. 业务交易（business transaction）

业务交易分为业务层面和技术层面两种定义。业务层面交易是指完成一次完整的业务操作，技术层面的交易是指进行一次应用程序至应用程序、或者应用程序至数据库的系统操作。

* 1. CPU利用率（CPU utilization）

系统或软件产品在运行过程中CPU的使用率。

* 1. 内存利用率（Memory utilization）

系统或软件产品在运行过程中内存的使用率。

* 1. 传输利用率（transmission-untilization ratio）

系统或软件产品在执行规定的数据传输功能时，传输吞吐率占传输设备最大传输吞吐率的百分比。

* 1. 构建环境

构建环境是指执行构建任务时所在设备的软件和硬件环境。

* 1. 构建任务

构建任务是指定义从输入到输出过程对构建环境、构建方法等必要因素的定义。

* 1. 构建过程

构建过程是指构建任务从开始执行到结束。

* 1. 构建策略

构建策略是指针对构建过程的特定结果设定的处理策略。

* 1. 复刻

复刻是指在软件开发中，开发者为一个版本仓库创建一份拷贝，并在拷贝上进行独立于源版本仓库的开发，从而分离出一个新的软件副本的活动。复刻不仅仅创建了一个分支，还将软件开发活动与源版本库分离开来。

* 1. 合并

是指当一个版本在多个独立分支中被修改后如何合并这些修改成为同一个版本的操作。

合并评审：是指在合并之前，允许多人对合并时所提交的变更内容进行代码审查，并给予合并意见的过程。

* 1. 基线

基线是版本仓库中每个版本在特定时期的一个“快照”。它提供一个正式标准，随后的工作基于此标准，并且只有经过授权后才能变更这个标准。建立一个初始基线后，以后每次对其进行的变更都将记录为一个差值，直到建成下一个基线。

* 1. 代码审查

是指对计算机源代码系统化地审查，常用软件同行评审的方式进行，其目的是在找出及修正在软件开发初期未发现的错误，提升软件质量及开发者的技术。代码审查常以不同的形式进行，例如结对编程、非正式的看过整个代码等。

* 1. 测试计划（Testing plan）

一个叙述了预定的测试活动的范围、途径、资源及进度安排的文档。它确认了测试项、被测特征、测试任务、人员安排，以及任何偶发事件的风险。

* 1. 测试脚本（Testing script）

一般指的是一个特定测试的一系列指令，这些指令可以被自动化测试工具执行。

* 1. 3.1 构建任务 Build task

作为一个构建软件的配置，以及流程。

* 1. 凭据 Credentials

保存用户保密信息，例如：用户名和密码。

* 1. 运行环境 Runtime environment

构建中运行的环境或者计算机

* 1. 源代码托管软件 Source code management software

保存源代码的软件

* 1. 执行记录 Build history

执行构建的信息，日志，以及结果。

* 1. 构建产出物 Build output

构建任务中生产的文件

* 1. 软件单元测试 Software unit test

软件中研发的自动化测试

* 1. 静态源代码扫描 Static code analysis

对源代码执行静态的分析，指出源代码质量(10.9)

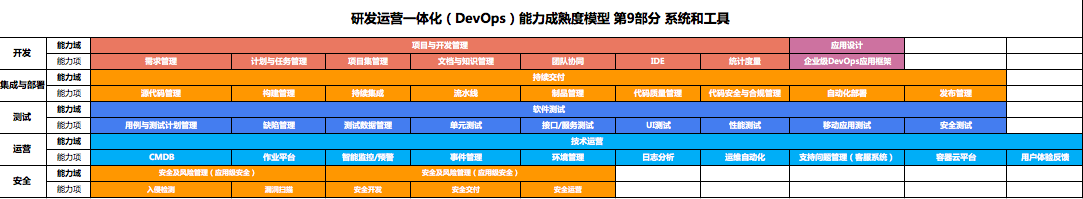
1. 缩略语

下列缩略语适用于本文件。

CI Continuous Integration 持续集成（样例）

CD Continuous Delivery 持续交付（样例）

1. 总体架构



1. 研发运营一体化（DevOps）系统和工具标准架构图

研发运营一体化（DevOps）能力成熟度模型中系统和工具部分包含开发、集成与部署、软件测试、技术运营和安全等的能力要求。

* 1. 项目与开发管理（名称是否变更为研发过程管理？是否专门增加单独产品管理模块？）

【建议统一项目、需求、任务的定义及彼此之间的关系，其中任务可包含开发任务、Bug修复任务等】

【仅有项目集管理，缺少项目管理】

补充：PMP定义中，项目管理指科学地组织人力在合理利用有限的资源，有限的时间内，按项目事先确定的要求，事先项目目标的努力的综合。



项目与开发管理是指xxxxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 需求管理

需求管理xxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 计划与任务管理

计划与任务管理是对于项目与开发过程中所有计划与任务的管理活动。应包含以下基本功能：

——开发计划管理宜支持看板协作项目计划管理与Scrum项目计划管理【仅描述看板协作与Scrum是否过于局限】

—— 【建议增加：任务需要与项目、需求、缺陷等关联】-ok

——支持迭代/里程碑规划，创建、修改、删除、查询

——支持面向任务及面向迭代的视图迭代规划

——支持展示按迭代/里程碑展示任务

——支持创建、修改、删除、查询任务管理

——任务参数字段宜包含：标题、类型、标签、描述、优先级、重要程度、处理人、抄送人、项目、状态、迭代、父任务项、计划日期、预计工时、实际工时、发布版本、附件、完成度。

——支持任务拆分子任务

——支持为任务添加关联任务

——支持任务指派、再指派、认领、二次认领

——支持记录任务的状态，如：未开始、进行中、已完成

——支持可视化的任务跟踪，如：看板

——支持任务列表的导入导出

——支持以树形视图或表格视图式呈现的任务项列表

——提供多维度的项目统计报表，展现项目宏观和微观进展

可以包含以下高级功能：

——支持【任务/】迭代/里程碑关联源代码对应分支，关联对应分支的CI/CD运行结果

——支持【任务/】迭代/里程碑自定义配置关联的资源，如文档文件

——支持设置迭代/里程碑的查看、添加、修改、删除权限【建议调整为统一功能】

——支持批量编辑任务、任务处理人及状态

——支持任务添加富文本信息

——支持设置任务参数显示列【建议更加清楚的描述】

——支持设置任务上传附件的数量与大小【建议删除】

——支持对任务进行评论

——支持按自定义条件筛选任务~~集~~【什么是任务集，且是否有必要将任务集群管理】-ok

——支持以看板、表格方式展示任务~~集~~

——支持以不同属性维度对任务集进行分组，排序后展示

——支持设置任务的查看、添加、修改、删除权限【建议调整为统一功能】

——支持任务配置状态工作流

——支持状态工作流设置状态变更权限

——支持任务变更/分派/逾期消息【通知】机制

——支持任务操作日志

——~~支持任务关联源代码~~【重复】

~~——支持任务关联相关代码的CI运行结果【重复】~~

~~——支持对任务集以自定义的范围、任务的各个维度、各种的聚合进行统计并展示报表【重复】~~

——支持用户自定义仪表盘，将用户关注的信息整合到一个页面上显示

——提供二次开发的API接口

——提供跨平台支持【请解释什么是平台，如何跨】-可能删除，PC端、大屏同步展示。

* + 1. 项目集管理

项目集管理对项目集运作过程中，项目、子项目集的运作情况进行管理的活动。应包含以下基本功能：

——项目集中可以包含多个项目和子项目集

——设置项目集PM，以及其他参与人

——设置以及展示项目集的进度、健康度【请详细解释设置的含义，由于与任务与计划管理中进度相关联】

——设置里程碑，分配负责人【项目集有里程碑吗？】

——录入需求、任务【建议调整为录入或关联需求与任务】

——填写及展示项目集的进度、健康度

——录入或关联需求任务

——记录跟踪项目集风险【请解释如何进行风险管理】

——展示它包含的所有项目或项目集的进展情况

——里程碑快到期发送消息通知给PM和里程碑负责人。

可以包含以下高级功能：

——项目或项目集，可以属于多个项目集；

——树形展示项目集中包含的项目和子项目集，子项目集可以逐层下钻；

——包含的项目和子项目集中的里程碑和需求任务等在同一甘特图中展示；

——根据里程碑进展、需求任务完成情况，以及包含项目和子项目集的进度和健康度，自动计算项目集的进度和健康度；

——汇总包含的项目和子项目集中的需求、任务、风险；【是否有必要？】

——包含的项目和子项目集中的风险，可以自行决定是否透出到项目集中；

~~——【董越】里程碑快到期发送消息通知给PM和里程碑负责人。（基础功能9.27）~~

* + 1. 文档与知识管理

文档与知识管理是对针对文档和知识的管理活动，应包含文档管理和知识管理两个部分。

* + - 1. 文档管理

文档是产品交付的核心研发资产之一。常见文档类型包括架构设计文档、用户帮助手册、系统原型文档等。常见文档格式包括微软Office WORD/PPT/EXCEL、PDF、WPS Office文档、JPG等格式的图片文档。

文档管理应包含如下基础功能：

——支持用户按照一定的目录结构对各种文档进行分门别类地管理

——支持通过~~拖拽方式~~【删除】批量上传本地目录或文件至文档管理服务中

——支持批量下载文档

——支持常见格式文档的在线预览

——支持文档版本管理，用户可以选择文档的某个版本进行过查看、下载等操作

——支持搜索功能，用户可以通过文件名或文件名关键字等快速查找到所需的文档

文档管理宜包含如下高级功能：

——支持本地与远端文件的同步【建议删除】

——支持回收站功能，以防止用户误删除文档

——支持文档的在线编辑及多人并行编辑

——支持对文件内容进行搜索

——支持共享功能，可以在不同用户之间进行共享【共享的含义比较模糊，是权限控制还是为分享？】

* + - 1. 知识管理

知识管理涵盖产品交付过程中个人或者团队的各种知识内容，例如会议纪要、版本ReleaseNotes、技术材料等等。知识管理应包含如下基础功能：

——支持词条创建、编辑

——支持多人协同编辑

——支持词条分享

——支持搜索

——可导出为常见文档格式（例如Word、pdf）

——宜支持Wiki格式

* + 1. 团队协同【这部分内容描述的就像一个APP，团队协同的定位到底是什么？】

团队协同是将同一个开发组织内不同角色、不同个体、不同阶段的资源整合形成协同效应，可以增加研发组织的信息透明度和信息流动速度，实现组织决策的高效和准确性。应包含以下基础功能：

——支持将研发工作过程、工作结果在组织内进行公开展示

——支持将项目规划、需求、工作项、代码、测试用例、缺陷、构建任务、部署任务、流水线任务等向指定人员开放展示

——支持不同角色基于工件进行协作

——支持多角色协作进行分析需求、工作项流转、代码合并、主流代码分支模型、代码评审、多语言并行构建、依赖包管理、镜像仓库等工作

——支持团队仪表盘、项目仪表盘、可视化看板、Wiki、文档管理、Git代码仓库、聊天室等团队协同工具，以将度量数据标准化一致呈现给所有角色和个体、分享知识经验、信息透明传递。

——支持记录所有人在平台上的所有操作，以回溯经验和教训，并满足审计合规

* + 1. 统计度量

统计度量是对DevOps过程的进度、质量、效率相关数据化指标展示。

应包含以下基本~~功能~~：

~~——进度相关指标【包含但不仅限于】：需求累计流图、缺陷趋势图、需求完成数、新建缺陷数。~~

——【董越】进度相关指标：包括但不限于需求累计流图、缺陷趋势图、需求完成数、新建缺陷数。

~~——代码内在质量相关指标【包含但不仅限于】：包括但不限于代码质量、千行代码bug率、缺陷Reopen率、测试通过率。~~

——【董越】代码内在质量相关指标：包括但不限于代码质量、千行代码bug率、缺陷Reopen率、测试通过率。

~~——交付外在质量相关指标【包含但不仅限于】：包括但不限于故障率、线上问题率、发布回滚率。~~

——【董越】交付外在质量相关指标：包括但不限于故障率、线上问题率、发布回滚率。

~~——需求交付时长相关指标【包含但不仅限于】：需求从提交到交付的时长。~~

——【董越】需求交付时长相关指标：包括但不限于需求从提交到交付的时长。

~~——缺陷解决时长相关指标【包含但不仅限于】：包括但不限于缺陷从创建到关闭的平均时长，表征解决缺陷的效率。~~

——【董越】缺陷解决时长相关指标：包括但不限于缺陷从创建到关闭的平均时长，表征解决缺陷的效率。

~~——代码交付时长相关指标【包含但不仅限于】：代码从提交到交付的时长。~~

——【董越】代码交付时长相关指标：包括但不限于代码从提交到交付的时长。

~~——人效相关指标【包含但不仅限于】：对使用人员的基本产出能力度量， 包括但不限于完成需求数、解决缺陷数、完成任务数、提交代码量。~~

——【董越】人效相关指标：对使用人员的基本产出能力度量， 包括但不限于完成需求数、解决缺陷数、完成任务数、提交代码量。

~~——【董越】统计度量的维度：项目维度的统计和度量。个人维度的统计和度量。（项目维度和个人维度等描述不好理解9.27）~~

~~——【董越】统计度量的维度【包含但不仅限于】：项目维度的统计和度量，即统计特定项目的进度、需求交付时长等相关指标。个人维度的统计和度量，即统计特定个人的代码内在质量、人效等相关指标。（项目维度和个人维度等描述不好理解9.27）~~

——【董越】统计度量的维度：项目维度的统计和度量，即统计特定项目的进度、需求交付时长等相关指标。个人维度的统计和度量，即统计特定个人的代码内在质量、人效等相关指标。

可包含以下高级~~功能~~指标：

~~——进度相关指标：迭代燃尽图、需求控制图。【这是展示方式不是指标】~~

——【董越】进度相关指标：通过迭代燃尽图、需求控制图等方式展示。

——需求交付时长相关指标：对需求每个阶段的度量，包括但不限于需求响应时长、需求分析时长、需求设计时长、需求排期时长、需求开发时长。通过各种时长，能够定位需求交付瓶颈角色，用于改进。

——代码交付时长相关指标：对交付上线的每个阶段进行分解。包括但不限于代码提交、构建、单测、集成、部署、集成测试时长、等待时长。能够定位和反交付过程中的各个步骤，用于改进交付过程。

——人效相关指标：人力分配和人力负载的度量。人力分配：包括但不限于每个项目的真实人员投入量。人力负载：包括但不限于项目并发度、需求并发度。

——统计度量的维度：组织维度的统计和度量，即从企业组织架构和团队维度提供整体的统计和客观的分析。【董越】~~应用交付过程的效率和质量透明。~~应用维度的度量，即应用交付过程的效率和质量透明。

* 1. 应用设计与开发
     1. 企业级DevOps应用框架

企业级DevOps 应用框架是指xxxxxx。应包含以下xx：

企业级DevOps 应用框架是指提供一套含服务发现、自动注册、负载均衡、开发组件、契约规范、高性能通信、公共组件、多语言支持等开发框架及支撑平台，以提升开发和运营效率。

应该包含以下基本功能：

（主要问题：应该描述软件框架的功能1101）

——服务发现：服务发现是对计算机网络上的设备和由这些设备提供的服务的自动检测，旨在减少用户的配置工作量。

——自动注册：服务注册中心可以给客户端提供可供调用的服务列表，客户端在进行远程服务调用时，根据服务列表然后选择服务提供方的服务地址进行服务调用。而自动注册是由服务实例负责在服务注册中心注册和注销，同时需要发送心跳来保证注册信息不会过时。

——负载均衡：负载均衡是一种计算机技术，用来在多个计算机（计算机集群）、网络连接、CPU、磁盘驱动器或其他资源中分配负载，以达到最优化资源使用、最大化吞吐率、最小化响应时间、同时避免过载的目的，用于解决互联网架构中的高并发和高可用的问题。

——开发组件：基于组件的开发是一种软件开发范型，通过复用已有的组件，软件开发者可以“即插即用”地快速构造应用软件。这样不仅可以节省时间和经费，提高工作效率，而且可以产生更加规范、更加可靠的应用软件。

——契约规范：契约式设计是一种设计计算机软件的方法。这种方法要求软件设计者为软件组件定义正式的、精确的并且可验证的接口，以生成规范、可靠的应用软件。

——高性能通信：

——公共组件（和开发组件重复1101）：基于组件的开发是一种软件开发范型，通过复用已有的组件，软件开发者可以“即插即用”地快速构造应用软件。这样不仅可以节省时间和经费，提高工作效率，而且可以产生更加规范、更加可靠的应用软件。

——多语言支持：语言无关规范是一种编程语言规范，它提供了一个公共接口，可用于定义适用于任意语言绑定的语义，以支持多种语言，减少或消除不同语言无法使用同一种应用的情况。

——支撑平台：支撑平台是一个信息的集成环境，是将分散、异构的应用和信息资源进行聚合，通过统一的访问入口，实现结构化数据资源、各种服务跨数据库的无缝接入和集成，提供一个支持信息访问、传递、以及协作的集成化环境，实现个性化业务应用的高效开发、集成、部署与管理。

+是否增加网关

~~（5.2.2 企业级DevOps应用框架~~

~~企业级DevOps 应用框架是指xxxxxx。应包含以下xx：1101）~~

* + 1. 云IDE（1101）

云IDE是指面向软件开发者的云端开发环境(定义不明确)。应包含以下基础功能：

+是否应当在描述列出使用场景（京东）

——支持在云端创建工作空间

——可自动生成用户工作空间

——支持在线编写代码

——支持用户定制化资源分配(表述不清)

——支持语法加亮、关键词自动补全及智能提示

——支持界面风格和多视图切换

——可自动关联技术栈库文件，快速生成软件包(表述不清)

——支持断点调测和环境变量设置

——支持访问后端环境，跨网络/跨容器联调

——支持主流技术栈，例如Java/Node.js/Android/.Net/C/C++/python/php等

——支持标准插件机制快速添加技术栈

——可与DevOps其他交付环节无缝集成（描述需要再准确些）

——支持与主流代码仓库对接，可实现代码自动下载、更新与智能提交（描述不清）

高级功能

——宜集成代码静态检查，~~快速发现低级源码错误~~

——宜集成代码review功能，支持团队成员检查逻辑及深层次问题

——可集成软件交付流水线，支持功能快速交付和缺陷快速修复

~~——可提供独立的本地IDE、第三方本地IDE插件或者Web IDE~~

——宜提供多种开发框架和运行环境的例程代码（描述不清）~~，支持用户快速入门和后续编码~~

* 1. 持续交付与持续集成（此章节篇幅过细，建议与其他部分描述的宽泛程度保持一致。）

持续交付xxx。应包含以下xx：

——xxxx

——xxxx

(应增加持续交付和持续集成的概念描述)

* + 1. 版本控制系统（Version Control Systems）

版本控制系统又称源代码管理系统，是指一种记录一个或若干文件内容变化，以便将来查阅特定版本修订情况的系统。

* + 名词解释
    - 复刻（Fork）：复刻是指在软件开发中，开发者为一个版本仓库创建一份拷贝，并在拷贝上进行独立于源版本仓库的开发，从而分离出一个新的软件副本的活动。复刻不仅仅创建了一个分支，还将软件开发活动与源版本库分离开来。
    - 合并（Merge）： 是指当一个版本在多个独立分支中被修改后如何合并这些修改成为同一个版本的操作。
    - 合并评审（Merge Requests）：是指在合并之前，允许多人对合并时所提交的变更内容进行代码审查，并给予合并意见的过程。
    - 基线： 基线是版本仓库中每个版本在特定时期的一个“快照”。它提供一个正式标准，随后的工作基于此标准，并且只有经过授权后才能变更这个标准。建立一个初始基线后，以后每次对其进行的变更都将记录为一个差值，直到建成下一个基线。
    - 代码审查（Code Review）：是指对计算机源代码系统化地审查，常用软件同行评审的方式进行，其目的是在找出及修正在软件开发初期未发现的错误，提升软件质量及开发者的技术。代码审查常以不同的形式进行，例如结对编程、非正式的看过整个代码等。
      1. 版本仓库（Repositories)

版本仓库又称源代码仓库，被用来存储源代码等版本文件，版本仓库应具备以下功能：

——创建版本仓库

——对版本仓库添加描述和标识

——浏览和查阅版本仓库

——复刻版本仓库

——归档版本仓库

——将版本仓库下载到本地

——将本地版本提交到版本仓库

——为版本仓库上指定成员，并根据可访问、可读取、可写入等不同能力将成员划分为管理员、负责人、开发者、协作者等不同角色

版本仓库管理可以具备以下功能：

——建立团队或路径，并在其中指派不同角色的成员，将版本仓库按一定模式归类存放

——更改版本仓库的可见性为全员共享或部分可见

——限制仓库总大小，限制仓库的单文件大小，对限制额度进行缩小或提额

* + - 1. 权限管理（Roles And Permissions）

版本控制系统的权限管理应当具备以下功能：

——应能在不同对象上（如团队、项目等）定义人员角色

——可定义的人员角色应当包含管理员、负责人、开发者、协作者等，并能将可访问、可读取、可写入等不同能力划分于不同的角色。

* + - 1. 分支管理（Branching）

分支管理应当具备以下功能：

——创建和删除分支

——浏览和查阅分支

——根据分支查看变更历史

——查看分支对应的版本文件

——能够在分支上定义合并策略，限制不同开发人员在分支上的合并权限

分支管理可以具备以下功能：

——定义分支创建规则，对分支的创建进行限定

——对分支增加描述和分类

——在分支上定义角色，指认分支的管理员，定义读取、写入和合并权限

* + - 1. 变更与合并管理（Committing And Merging）

变更管理应具备以下功能：

——将本地修改提交到版本仓库，完成变更

——查看某个变更的当前全部版本文件，并将版本文件拉取到本地

——查看一次变更或多次变更的文件差异

——对比两个不同变更之间的文件差异

——查看变更的历史记录以及变更之间的关联关系

——将某个或多个关联变更回滚至之前状态

——将某个变更标识为Tag，并作为基线管理

——变更被提交时，提供Hooks用于出发自动化集成和检测

——将一个或多个变更创建为合并评审，用于合并到待合并分支，并可设置经过评审后才可合并

——查看合并评审的历史列表，以及每个合并评审引入的变更

变更管理可具备以下功能：

——查看变更与分支的关联关系

——针对变更进行讨论

——合并评审可根据代码扫描、持续集成与自动化测试系统反馈情况，阻止有风险的版本合入

* + - 1. 基线管理（Baseline Management）

基线管理应具备以下功能：

——查看基线列表

——从指定的变更创建基线

——将本地创建的基线同步到版本仓库

——在版本仓库中删除基线

——查看基线对应的版本文件，并能将对应的版本文件下载到本地

——查看不同基线间、变更与基线间、分支与基线间的文件差异

基线管理可具备以下能力：

——能根据版本仓库的成员权限对基线的创建进行限制

——能在版本仓库定义基线的命名规则，避免不符合规则的基线创建

* + - 1. 代码审查（Code Review）

代码审查应具备以下能力：

——能够对变更和待合并的变更发起审查，邀请审查人查看相应的版本差异

——能够对差异的代码行进行标注和讨论

——审查人能够在不同时间和地点浏览审查，并给出意见

代码审查可具备以下能力：

——能够对代码审查进行数据度量

——代码审查流程通知可与主流的即时通讯服务或团队协作工具集成，而提供实时的消息通知

——代码审查流程能够在移动终端进行查阅、讨论并完成相关审批

* + - 1. 存储和备份（Storage And Backup）

版本管理系统应对版本仓库提供稳定可靠的存储和备份方案：

——同一个版本仓库能够支持多份存储写入，当存储故障时可切换备份存储

——提供实时和定时的备份机制，可使用备份还原版本仓库

存储和备份可具备以下能力：

——同一个版本仓库能够提供多地存储

——在有多地访问需求时，能提供就近访问

——大量复刻并修改同一版本仓库时，能让存储容量不随复刻数线性增长

——支持使用通用的文件存储系统（如x86平台的存储系统）

* + - 1. 安全保障（Security Management）

版本控制系统应提供安全保障能力：

——应唯一确定访问账号身份

——保证无法越权访问

——能够保留访问记录，用于查阅回溯

——应保留危险操作记录，用于审计回溯

——应支持如https / SSH等加密传输方式，保证传输过程的网络包被截获后无法轻易解密。

* + - 1. 集成能力（Integrations）

版本控制系统应具备与需求敏捷、持续集成、持续交付、代码检测、自动化测试等工具集成的能力：

——应为变更、合并和评审提供Hook，用于触发自动化流程

——应提供CI/CD中基本功能的API，用于其他系统调用接入

版本控制系统可具备以下集成能力：

——支持OAuth等安全的三方认证能力，保证需求敏捷、持续集成、持续交付、代码检测、自动化测试等工具在访问版本资源时，访问权限受到当前操作人角色权限的限制

——提供研发生命周期中所需功能的API，用于其它系统深度集成

* + - 1. 其他能力（Others）

——可提供项目维度和团队维度的动态墙用于汇聚研发过程有关的消息和事件，使用会话的形式聚合研发过程中发生的事件与关键结点，并可使用移动终端查阅。

——会话墙能力可允许需求敏捷、持续集成、持续交付、代码检测、自动化测试等工具调用。

* + 1. 构建管理

构建是指从代码或其它产物构建出指定输出物的过程，构建管理是指管理以上过程涉及的任务、过程、策略和环境等必要因素。

应具备以下基本功能：

——通过配置模板或脚本初始化构建环境。

——通过配置模板或脚本定义构建任务。

——初始化环境和构建任务的配置模板或脚本的存储有版本管理，可配置标签。

——构建过程的输出可重定向到文件或日志服务，能够持久化存户和查询。

——构建过程可以设定执行时间，针对超时的情况可设定超时策略。

——构建过程的输出物设定输出地点、传输协议，针对输出失败的情况设定失败策略。

——构建过程的触发支持通过 HTTP RESTFUL 、GRPC 等多种方式触发任务执行。

——构建过程的结果可设定多中通知策略，支持 Email、IM 等。

应具备以下高级功能：

——管理构建资源，针对构建任务设定优先级进行调度。

——能使用分布式系统管理构建任务。

* + 1. 持续集成

持续集成是指通过源代码构建软件的流程。应该包含以下基本功能：

——保存多个构建任务

——设置代码仓库地址，以及拉取源代码的凭据

——设置一个或多个构建命令

——支持多种源代码语言的编译

——支持多种源代码托管软件

——设置自动触发条件：定时触发，源代码变更触发

——构建任务应该含多个执行记录

——构建执行记录展示记录状态，以及结果

——构建执行记录展示构建过程产出的日志

——支持构建命令中执行软件单元测试

——支持执行静态源代码扫描

——支持定义构建执行环境

可以包含以下高级功能：

——可以支持提供执行持续集成依赖的服务，例如：数据库服务，缓存服务等

——构建执行记录中可以保存构建产出物

——构建执行记录中可以保存构建报告，单元测试报告，测试覆盖率报告

——可以保存构建产出物到第三方软件

——可以支持在一个构建任务中多个源代码仓库的

——可以支持定义构建任务超时时间，并在执行过程不允许

——支持构建容器镜像，以及推送构建的容器镜像到第三方软件

——支持源代码缓存

——支持触发其它构建任务，或者其它第三方软件

——支持定义执行超时时间

——建议在高级功能中添加：

——支持源码多分支管理策略的构建场景

——支持动态可伸缩的构建执行环境

——支持构建结果通知（及时消息、邮件等）(10.9)

* + 1. 流水线

流水线是将代码提交到部署上线整个过程中的自动化实现。应包含以下基础功能：

——支持集成调度代码托管、代码检查、编译构建、部署、测试、发布等任务，应实现提交代码后的端到端的产品自动化交付与发布

——支持业务流程按需定制，可选择要执行的步骤和顺序

——支持级联调度与分层分级(是否增加描述1101)

---增加串行并行能力的描述（1101）

——支持按时间计划或代码提交等多种触发方式

——支持流水线状态可视化，宜清晰呈现价值流动，宜及时准确定位问题

——支持参数配置

++增加质量管理控制,支持多种条件判断

流水线宜包含如下高级功能：

——支持人工审批

* + 1. 制品管理

专家建议：增加制品版本控制模块；另外是高级功能和基本功能划分的问题。

制品管理是对软件研发过程中生成的产物的管理，一般作为最终交付物完成发布和交付。 制品即构建过程的输出物，包括软件包，测试报告(存疑,招商银行,是否作为元数据)，应用配置文件(是否适合作为制品,应当能够作为元数据1101)等。 比如jar包、zip包、rpm包、docker镜像等。

应包含以下基本功能：

--(缺少制品添加方式的描述(谐云科技)1101)

——支持至少一种制品类型；

——为制品添加版本信息等元数据信息(元数据信息希望能细化,烽火科技1101)；

——使用制品的审计日志；

——基本的权限管理；

——代理其他制品库；(不应该作为基本功能,应作为高级功能,一些企业无法实现代理能力,平安银行1101)

——检索制品；

——备份和恢复

制品管理高级功能如下：

——支持多种制品类型；

——制品分发支持大规模并发、全球加速；(应当增加对场景的限制1101)

——与持续交付系统高度集成；(描述有欠缺,是否和API一行合并谈集成能力1101)

——基于元数据信息实现制品的安全管控、晋级(晋级描述不明1101)

是否应加入制品的安全熔断机制,当包不安全时快速熔断阻塞依赖(华三1101)

——REST API支持；

——通过关键字、坐标数据、CheckSum等多种方式检索制品

——基于角色的权限管理

——基于自定义策略的访问控制

——制品文件安全扫描

——分析和查看制品依赖关系

——基于CheckSum的制品去重存储；

——自定义制品清理/归档策略(是否应该改为基本功能,平安银行,征求下云产品的建议1101)

--(是否增加配额管理能力作为高级功能1101)

——支持多种存储源，(建议是写支持存储类型,不应出现具体厂商1101) 比如阿里云OSS、AWS的S3等

* + 1. 部署管理

专家建议：生产环境的管理还是测试环境？此处是生产管理的，专家建议包含生产环境和测试环境。

(增加对测试等各类环境的描述1101)部署管理是DevOps持续交付中的一个重要环节，它借助自动化工具与统一规范，实现仓库版本到生产环境的自动化部署，提升开发和运维人员快速部署的能力。应该包含以下基本功能：

----是否应该有 权限管理（1101）

——部署工具应具备批量（海量）并发部署的能力；

——支持灰度、全量(是否是发布管理的概念,而不是部署管理1101)等部署方案的实施，实现对生产环境的平滑升级；

——部署前支持备份上一版本的生产环境，在当前版本出现问题后能支持版本快速回退；

——支持业务配置文件的自动生成与下发以及统一管理；

——当业务架构调整后(存在疑问,mark1101)，部署方案(范围较大1101)仅需调整布署节点配置项即可(布署节点指的是什么,京东云1101)；

——部署工具与业务架构松耦合，无需业务因布署方案或工具做任何改造；

——部署对象服务器支持windows、linux等主流操作系统；

——部署对象服务器在CMDB集中管理，与部署工具联动实现任意组合集群的版本部署；

可以包含以下高级功能：

——部署工具具备调度编排的能力，能同时对不同功能模块版本实现一次性全流程部署；

——支持上下文传输部署，即上一部署节点的输出能作为下一节点的前置条件或输入参数；

* + 1. 发布管理

(如何描述对业务影响) 发布管理是指一个应用或多个集成应用程序从测试完成到生产环境用户可见的发布流程(定义混淆,如何和部署区分,华为1101)。应包含以下：

应包含以下基本功能：

——发布计划，规划软件程序的整体发布计划，包含但不限于：谁在何时、何地、怎样、发布了什么。以及对比发布计划和发布状态的能力。

——发布定义，定义发布流程和审批工作流。

——发布策略，通过选择进行发布的目标环境、超时时间、暂停点、灰度发布等发布的具体策略，执行对应的发布动作；发布策略包含并不限于一键式发布，零停机发布、灰度发布、金丝雀发布等

----(与布署管理关联1101)

——发布执行，自动化地执行发布策略，如策略中有暂停点，应验证后继续执行发布

——发布确认，通过发布规划中软件发布的确认点，进行发布确认，如与预期不一致，可快速回滚到发布前的软件版本

可以包含以下高级功能：

——发布决策，实现个角色的协同，评审决策，控制应用版本向后续环境的流动

——风险评估与质量门禁，提供发布影响分析视图，可快速查看发布产品的发布风险，在关键节点控制版本质量，支持自动质量门禁

——发布度量，发布报告视图，提供发布的状态报告与历史发布数据

* + 1. 环境管理

环境管理是一种配置管理活动，确保应用在多个环境之间达到持续交付的目的。应该包含以下基本功能：

——可以定义不同的环境类型（开发、测试、预发布及生产环境）；

——可以定义不同的环境依赖资源信息及其配置，比如主机、容器集群、DNS、中间件、其他基础设施服务等等；

——可以根据环境的配置快速生成交付(交付一词不妥1101 ) 环境；

——可以让环境的配置信息存储在构件库中，版本化控制配置信息；(构件库的含义不清，应当澄清)

——可以支持应用运行的环境是静态主机集群或者是动态的容器集群；

——可以支持不同的应用有不同的基础设施及服务依赖；

——可以支持不同的对象分块构建，比如说构建基础设施、构建中间件或者操作系统环境等等；

——可以支持不同的环境采用不同的构建技术，比如说虚拟化、容器等等，但测试环境和生产环境必须类似；

——可以支持环境的配置信息与应用或者项目关联；

——对环境提供监控功能；

可以包含以下高级功能：

——提供与不同的配置管理工具对接功能；

——提供与不同制品库对接功能，比如说maven仓库和容器仓库等等（1101）；

——提供开放式的API供外围平台编排调用；

——提供所有环境的变更日志和审计功能；

——提供环境配置信息的权限控制；

——提供测试验证的工具来验证环境的有效性；

* + 1. 数据管理（待定）

数据管理xxxxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 应用配置管理

应用配置是指应用启动或动态运行时影响应用行为的配置项，典型例子如数据库连接串(含用户名密码)，本地服务线程池数，RPC连接超时时间设置，三方软件LicenseKey，等。应用配置管理，是对应用配置进行集中式管理的系统和方法。采用应用配置管理，可以让配置发布有效解耦程序开发，程序发布等流程，并让程序能动态响应配置的变更，提高DevOps效率。

应包含以下基本功能：

——配置的最小粒度应对应单一配置ID，其内容应不限于KV，Json, XML等格式，用户应能自由配置。

——配置内容大小应有限制，但是不宜太小，如应超过100KB以上。

——配置应能基于租户，分组，或其他不同粒度规则进行隔离。

——对配置的访问应有鉴权，以控制账号对相关配置项的读写权限操作。

——应提供相应程序接口供应用程序实现读、写配置的功能。

——应提供相应程序接口供应用程序实现配置订阅的功能，如配置变更，则程序可在短时间内做出响应。

——应用程序应可以基于各类语言如java, go, node, c++等多语言SDK来操作配置。

——应用配置管理应能和各类CI/CD工具集成，如github，等。

可以包含以下高级功能：

——配置发布时应支持灰度发布，如根据某类机器的IP或某类机器的Label。

——应提供配置订阅监控，可查看配置被哪些机器订阅。

——配置应能区分版本，不同版本之间能一键回滚，以降低配置发错的风险。

——配置管理应能充分集成各类加密工具，如AWS KMS，Aliyun KMS，Hashcorp Vault，（不应出现具体产品1101）以保证配置安全性。

——配置管理中心以成为系统单点，如管理中心宕机，应用应能正常运行。(但新配置不能发布)

* 1. 测试管理

测试管理xxxxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 用例与测试计划管理

专家建议：批量添加改到基础功能中，效率的问题，重置测试计划多少天，建议执行效率加到高级功能。

用例与测试计划管理是对用例与测试计划的管理活动，包含用例管理和测试计划管理两大部分。

* + - 1. 用例管理

用例管理是对用例集、子用例集和用例的管理活动。

应该包含以下基本功能：

——用例集中可以包含多个用例和子用例集

——树形展示用例集中包含的用例和子用例集，子用例集可以逐层下钻

——设置用例的测试步骤，包括步骤描述、输入测试数据，期望结果

——用例支持不同的优先级、重要程度。

——可以区分手工和自动化用例

——用例集、子用例集和用例支持标签分类

——支持用例集、子用例集、用例和版本、需求、特性、故事关联

——支持自动化测试用例和测试脚本关联

可以包含以下高级功能：

——用例或子用例集，可以属于多个用例集

——支持在用例上添加附件

——不创建测试计划情况下用例可以随机执行

——可以从用例关联到测试计划

——支持对版本、需求关联用例的统计分析

——支持对用例执行情况的统计分析

* + - 1. 测试计划管理

测试计划管理是对测试计划的管理活动。

应该包含以下基本功能：

——支持测试计划创建和变更

——可以将用例集、子用例集和用例添加到计划以关联到用例

——可以按照标签批量添加用例集、子用例集和用例

——测试计划可以支持多轮次执行

——计划执行可以设置测试版本、测试环境、测试周期

——计划执行支持不同用例给不同测试人员执行，可以是指派或认领方式

——计划执行测试人员可以反馈用例的测试结果，自动给出计划执行进展和结果

——测试用例可以反馈失败原因

——失败原因是版本故障的用例，可以关联到缺陷

——可以对测试计划执行情况统计分析

可以包含以下高级功能：

——测试计划可以直接从需求自动添加关联的测试用例

——支持测试用例、环境的？（1101）

——用例执行测试时可以针对每个测试步骤测试并反馈结果

——自动化用例可以驱动测试工具执行脚本并自动反馈测试结果

——自动化用例驱动测试工具执行后可以自动分析用例的失败原因

——测试计划执行自动化用例对接测试工具具有可扩展性

——失败原因是版本故障的用例，可以自动向缺陷管理系统提交缺陷

——可以基于版本、需求对测试计划执行情况统计分析

——可以对测试计划执行效率进行统计分析

* + 1. 缺陷管理

专家建议：严重程度放到基本功能里面。-已改

缺陷管理是指在软件生命周期中识别、管理、沟通任何缺陷的过程，确保缺陷从被识别到解决关闭的过程被跟踪管理而不丢失。应包含以下基本功能：

~~——【董越】标记缺陷优先级；~~

——【董越】标记缺陷优先级和严重程度；

——将缺陷指派给特定的人；

——标记缺陷的不同状态，状态覆盖从新建到解决关闭的整个过程。

——缺陷可以关联截图、视频等；

——可区分缺陷类型；

——可按解决者、作者、优先级严重程度、当前状态等字段过滤；

——缺陷可关联到需求和用例；

可以包含以下高级功能：

——可添加评论；

——指派缺陷、更改缺陷状态，发送消息给相关人员；

——缺陷可关联到修复该缺陷的代码。

——可过滤未关联需求或用例的缺陷。

专家评审：

基本功能：

1. 文本框不需要具体讲
2. 缺陷可以关联截图、视频等
3. 过滤建议放到基本
4. 建议增加 记录缺陷关联到的需求和用例

5、建议增加 缺陷分类属性？

高级功能：

1. 权限控制属于通用的，是不是需要单独提出？
2. 高级5建议去除？

3、可以过滤未关联用例和需求的缺陷；

* + 1. 测试数据管理

测试数据管理指的是针对测试过程中所使用和产生的数据进行的管理。应包含以下基础功能：

——支持基于CSV、关系型数据库、文档数据库等类型的数据源提取生成测试数据

——支持录制接口请求和响应生成测试数据

——支持定义测试数据的元数据信息

——支持创建和编辑CSV形式的测试数据

——支持把CSV、关系型数据库、文档数据库等作为外部存储库存储测试数据

——支持设置规则批量生成测试数据

——支持使用脚本语言批量生成测试数据

——支持按照指定逻辑提取测试数据

——支持按照指定逻辑转换测试数据

——支持按照指定样式展示测试数据

——支持发现、清洗、保密敏感数据

——支持测试数据回滚

——支持分环境管理测试数据

——支持手工测试、自动化测试、Mock引用测试数据

专家评审：

1. 建议不独立成章，可放到环境管理中；
   1. 自动化测试

测试数据管理xxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 静态代码检查

静态代码检查是持续交付流水线中的一个重要环节，利用商用/开源/自研的代码检查工具在开发阶段发现缺陷，让缺陷在最短路径闭环。

应该包含以下基本功能：

——对代码进行静态扫描，发现代码缺陷、安全漏洞及编程规范、重复代码、复杂度高等代码坏味道问题

——能够自动触发/立即分析/定时开展，实时展示扫描进展状态，及时反馈代码检查结果

——方便查看告警及错误代码片段，提供规则描述及告警修复指导

——检查结果有优先级/严重程度的划分，能跟踪到状态

——支持检查规则配置，支持单个告警/批量告警/告警路径屏蔽等功能

——自动生成代码检查报告，有新增/修复/遗留告警等质量度量指标

——多种工具检查结果能够整合展示在报告中便于开发团队修复

可以包含以下高级功能：

——能够开展增量代码扫描，自动识别问题引入者

——能够集成进IDE中在编码更前端开展检查

——根据项目实际场景支持检查规则定制

——支持和缺陷管理系统打通，具备跟踪和驱动修复的能力

——代码检查报告中自动生成新增/修复/遗留告警趋势量化展示项目代码质量状态

——代码检查工具支持集成到持续交付流水线中自动运行实施

——支持配置项目代码质量符合度标准，在流水线中自动化实施，作为下一个环节的准入条件

* + 1. 单元测试（1101）

专家建议：测试应该执行方来做的，这里所描述的所有测试是执行测试还是包括了管理部分，比如单元测试覆盖率大于70%

单元测试是项目的开发质量管理活动。必须包含以下基本功能：

——自动化、独立性、可重复执行的特点（是单元测试特性）

——每个CASE有assert 来验证

——核心业务、核心应用、核心模块的增量代码确保单元测试通过（不是成熟度，建议去掉）-建议保留

——单元测试的CASE有手工编写和自动化生成2种~~，文件名要区分~~

~~——每个单元测试CASE执行时间不超过1秒可以指定。~~

——单元测试的报告中必须有分支覆盖率，CASE总数，成功执行总数，失败总数，异常总数，执行跳过总数

——单元测试和源代码用目录结构区分开

~~——使用单元测试框架~~

~~——单元测试有修改记录~~

~~——单元测试有评审记录~~

可以包含以下高级功能：

~~——使用单元测试框架~~

——使用MOCK

——每个单元测试CASE执行时间~~不超过1秒~~可以指定。

~~——单元测试有修改记录~~

~~——单元测试有评审记录~~

~~——单元测试的参数范围包含开发和测试共同确定（建议修改描述，要体现出能力项要求）~~

——使用工具自动化生成单元测试

——单元测试支持并行运行

专家评审：

* + 1. 基本上是成熟度描述，产品特性少，修改再评接口/服务测试

接口测试是通过直接在消息层向测试软件接口，确定功能、性能、可靠性、安全等是否满足预期的软件测试方法。REST风格接口是Web应用前后端和微服务之间主要的接口形式。

应包含以下基本功能：

——支持针对REST风格接口的测试

——支持HTTP、HTTPS协议GET、POST、PUT、DELETE等主要方法的测试

——支持基于Swagger或Open API文档辅助编写接口测试逻辑（建议放到高级）

——支持基于录制的接口请求和响应生成测试逻辑（建议放到高级）

——支持定义HTTP请求的URL、请求头、请求报文

——支持表单、文本、JSON、XML等多种格式的请求报文

——支持Basic认证、Token认证等接口鉴权认证方式

——支持针对HTTP响应的响应报文、响应头、响应码定义测试断言

——支持提取HTTP响应的内容，基于上下文创建后续HTTP请求

——支持循环、判断、暂停等测试逻辑 （建议描述为接口测试流程控制）

——支持定义不同测试环境的测试参数

——支持测试关键字，以复用测试逻辑（术语中增加关键字驱动，复用测试逻辑建议作为单独特性）

——支持测试套件编排测试用例串行顺序执行或并行执行（可能不是接口测试专有特性）

——支持调试测试用例，以验证测试逻辑（可能不是接口测试专有特性）

——支持记录测试执行日志和结果（可能不是接口测试专有特性）

——支持定义接口Mock，以解耦开发和测试依赖（目的建议去掉）

——支持流水线调用接口测试实现持续集成和持续测试

——支持接口测试覆盖率和测试通过率统计

专家评审：

1. rest风格描述过细，其它接口测试没有提到；
   * 1. UI测试（阿里董越老师1025之后改--1031版本）

UI测试是用户界面测试，英文名为User interface testing的简称。它通过浏览测试对象可正确反映业务的功能和需求，这种浏览包括窗口与窗口之间、字段与字段之间的浏览，以及各种访问方法 （Tab 键、鼠标移动和快捷键）的使用，且窗口的对象和特征都符合需求。

它应该包含以下基本功能：

~~——模拟器或真机浏览器打开被测对象（表述方式局限到web）~~

——【董越】在实际运行环境或模拟器中打开被测对象

~~——模拟用户的键盘、鼠标操作（建议采用人机交互的方式描述 ）~~

——【董越】模拟用户通过键盘、鼠标等进行的人机交互操作

~~——被测流程的顺序执行（必然的，可不描述）~~

~~——内容（文案、数值等）检查（被测对象属性检查）~~

——【董越】被测对象的内容和属性的检查

——测试步骤局部修改

——测试过程用例管理

——测试过程的本地回放、执行

可以包含以下高级功能：

——测试过程局部调试

——自动录制测试过程

——参数化测试过程中相应数据

——被测对象相应内容（文案、数值等）支持数据库对比

——重复过程的公共方法

~~——多环境执行（建议解释云端执行和多环境执行）~~

——【董越】支持一套脚本在不同环境(如本地开发环境、云端测试环境等)中执行

——~~云端执行~~

——【董越】支持对云端执行的测试任务的管理

——定时执行

~~专家意见：~~

~~建议作者线下再评审~~

* + 1. 移动应用测试

移动应用测试是指针对市场主流移动端（Android、iOS、H5、小程序等）利用自动化技术进行功能性和各项非功能性专项测试。包含适配兼容测试、移动自动化测试、客户端性能测试三大类测试。

5.5.5.1 适配兼容测试

适配兼容测试利用移动终端真机实验室，移动应用需要在各种参数搭配的机型上进行测试,以确保移动应用兼容用户使用的手机机型, 最大化客户群体。它应该包含以下基本功能：

——按需的机型选择

1. Android/iOS操作系统版本
2. 主流厂商的定制ROM
3. 屏幕分辨率
4. 硬件配置（CPU、GPU、内存）
5. 市场Top用户占比机型
6. 支持自选机型

针对H5

1. 各大厂商的浏览器和内核
2. WebView（如微信）

——安装、拉起、Monkey遍历、卸载

——适配问题的检测能力（崩溃、无响应、安装失败、拉起失败、卸载失败、UI异常、Exception）

——测试执行步骤的还原（操作和截图）

——基础性能指标采样（安装时间、启动时间、CPU、内存、流量、FPS）

——系统日志的收集

可以包含以下高级功能：

——用户账户自动登录

——支持运行主流自动化框架（如Appium、UIAutomator、XCTest）和自定义框架编写的兼容测试脚本（主流自动化框架，建议举例或修改描述，安卓官方、或苹果官方？此处是否体现自定义框架？已修改）

——支持自定义日志上报和导出

——支持API接口提交任务和接口文档

——能够对出现具体问题的设备进行远程调试

5.5.5.2 移动自动化测试

移动自动化测试，英文名为Mobile Test Automation的简称，它是将移动应用作为测试对象，把以人为驱动的测试行为转化为机器执行的一种过程。通常，在设计与编写用例脚本后，由测试人员上传至移动终端真机实验室自动执行测试，得到实际结果与期望结果的对比评估。在此过程中，达到节省人力、时间或硬件资源，增强测试深度，提高测试效率等效果。

它应该包含以下基本功能：

——支持通用的自动化测试框架

——模拟用户对应用的操作（点击、滑动、长按、轨迹操作、文本输入等）

——基本的识别能力（标准控件识别）

——测试信息管理能力（用例、日志、缺陷、报告）

——移动设备的管理运维能力（故障等重连、重测机制，设备稳定性）

可以包含以下高级功能：

——支持用户修改过的开源框架

——提供插件集成到开发IDE中

——提供丰富的Open API（包含设备信息获取，筛选，提测，结果拉取等灵活，详细的API能力）

——在云端设备上进行脚本调试

——进阶的识别能力（OCR识别，UI识别，游戏引擎层控件识别等）

——脚本录制及精准回放能力

——任务运行可视化及灵活干预能力（以单台设备为粒度进行终止、重测等干预操作）

——性能数据采集调试能力（包含FPS、内存等通用性能，以及引擎层的性能）

——云端设备灵活配置和高效调度能力

5.5.5.3 客户端性能测试

客户端性能测试利用移动终端真机实验室，通过在不同参数搭配的各档机型上利用性能测试工具获得各项性能指标，定位性能瓶颈和问题根本原因，提升移动应用的性能表现，以保障用户获得流畅、稳定的最佳使用体验。它应该包含以下基本功能：

——机型性能天梯（机型分档）

——基础性能指标采样（安装时间、首屏时间、响应时间（启动、滑动、界面切换等）、CPU、内存、流量、FPS、耗电量、温度、IO)-客户端必要的指标

——测试执行步骤的还原（操作和截图）

——系统日志的收集

——测试报告的导出

可以包含以下高级功能：

——支持运行主流自动化框架（如Appium、UIAutomator、XCTest）和自定义框架获取性能数据（跟前面的主流自动化框架描述问题一样已修改）

——支持版本对比、竞品对比

——支持API接口批量获取性能数据

——过度绘制检测

——支持性能问题根因分析

——支持多性能指标组合分析（建议高级增加 支持性能指标组合分析已添加）

——针对游戏提供unity3d,unreal 4引擎的深度性能数据

——针对H5提供瀑布流视图，连接视图，提供实时抓包、修改功能

专家评审：

1、建议高级增加 支持性能指标组合分析

* + 1. 性能测试

应该包含以下基本功能：

——支持性能测试项目的测试脚本、测试结果、测试报告的基本管理功能；

——支持主流测试协议；

——支持负载参数集读取数据文件功能；

——支持负载参数集自动生成序列数字、随机数字等功能；

——负载参数集数据读取支持：顺序、随机、数据文件读取、数据文件分段读取、文件读取的功能；

——支持脚本逻辑控制功能，脚本编辑功能；

——支持性能测试结果数据输出，包括测试发送数据及服务端响应数据等；

——支持思考时间设置功能；

——支持请求超时、响应超时设置功能；

——支持性能测试执行过程中服务端回送数据正确性检查的功能；

——支持长连接、短连接设置；

——支持数据上下文关联的功能；

——支持性能测试场景设置功能，性能场景：性能测试过程中模拟真实用户的服务流程或业务处理过程的一系列动作的集合；

——支持性能测试指标数据实时输出的功能；

——支持性能测试报告查看及导出功能；

——支持采集性能监控指标的功能；

——支持性能测试过程中各类错误显示、汇聚的功能；

——HTTP、HTTPS协议应支持GET、POST方法的测试；

——支持HTTP协议COOKIE设置；

可以包含以下高级功能：

——当分布式压力源未用尽时，系统支持多个测试用户同时调用压力源发起性能测试直到压力资源满载；

——支持多种协议的性能测试；

——支持自定义协议的性能测试；

——支持测试数据集可通过脚本或程序灵活生成；

——支持从数据库中自动读取负载参数集的功能；

（这类需求的适用场景相对狭窄，且产品自研的实现成本非常高，我们对它是否需要写进一个国家标准存疑）

——支持灵活配置、快速扩充压力源的功能；

——支持测试节点调用互联网公有云资源、局域网私有云资源、或其它可用的虚拟及物理资源；

——支持相同测试场景执行报告比对功能；

——支持采集各种性能指标的功能，如：网络通信指标、中间件指标、数据库指标等；

（如果全部都要采集实现成本非常高，且系统本身已经有替代功能。是否有必要写进国标？

如果坚持要写，那么希望可以列清楚需要具体需要采集哪些）

——支持可配置的性能测试挡板。性能测试挡板：在跨系统测试中由于客观原因等无法模拟全部测试流程中的非主要测试系统资源，故通过软件或程序来模拟其他相关系统的操作，该软件或程序称之为挡板，主要目的是保证测试流程的完整性；

（希望可以把“挡板”的解释放进“术语定义”章节）

——支持集成外部工具进行性能分析；

（是进行“性能测试”还是“性能分析”？如果是性能分析，主要是指哪个层面？）

——HTTP、HTTPS高级功能支持OPTIONS、HEAD、PUT、DELETE、TRACE、CONNECT方法的测试；

(从经验来看，用户较少用这些方法，是否需要都加上？用户基本用不上，且实现这个功能没什么技术难度，对于产品来说，这个应该算不上加分项。）

——HTTP、HTTPS、Socket协议支持脚本自动化录制、回放功能；

（加列了socket协议的原因是？如果加上socket，那TCP、PB的录制是否也需要加上？）

* 1. 安全开发

安全开发xxxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 代码安全与合规管理

代码安全与合规管理是为了在代码阶段拦截软件开发过程中可能会引入的安全漏洞而开展的管理活动。应包含以下基本功能：

——对代码进行扫描，检测出代码开发过程中引入的安全漏洞；

——代码扫描过程，基于OWASP和CWE的规范进行漏洞的检测；

——需要识别出有已知安全漏洞的依赖包；

——需要识别出有已知安全漏洞的开源软件；

——建立优选开源软件版本库，检测出非优选开源软件；

——检测二进制包/字节码的安全漏洞 ；

——展示检测出来的漏洞，明确告知漏洞来源，以及修改建议；

——建立漏洞管理追踪系统，跟踪误报，已修复，和未修复的漏洞；

——通过邮件，短信等信息，把检测结果通知责任人。

* 1. 安全交付

安全交付xxxxx。应包含以下xx：

——xxxx

——xxxx

* + 1. 安全测试

安全测试是通过分析检测对象业务流程，借助流量抓取、协议分析或手工检测等方式，以发现检测对象业务逻辑层面可能存在的安全漏洞和设计缺陷。应包含以下：

安全测试应包含以下功能：

——安全测试启动

* 展示所使用的业务信息；
* 涉及权限控制；
* 与后台有交互的功能；
* 需要引入的外部资源(包括但不限于图片、视频等静态文件以及外部链接等)；
* 可能涉及的第三方程序/代码的使用或更改；
* 其他可预见的安全风险。

——自动化安全测试

* 1)白盒扫描：基于应用的源代码进行安全审计，给出疑似安全风险点及其解决方案；
* 2)黑盒扫描：基于线上流量产生的接口进行漏洞扫描。扫描时对线上服务发送拼接了payload的畸形请求，以检测是否包含对应的安全漏洞。
* c)手工安全测试
* 包括但不限于xss、csrf、utf、url重定向、敏感数据、平行权限、文件上传、sql注入。测试过程中可借助抓包工具进行验证。
  1. 安全运营（腾讯云-待定）

安全运营的产品，是指系统上线运行中的安全工具。应考虑以下要求：1、评估系统安全风险，输出系统安全运营需求；2、根据需求，部署安全产品，包括几类：1）、主机安全；2）、数据安全；3）网络安全；4）应用安全；5）数据安全；6）访问控制安全产品，例如堡垒机。

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_