ICS 35.100.05, 35.240.40
L79, A11

T /IA C

中 国 保 险 行 业 协 会 团 体 标 准

T/IAC XXXXX—2018

保险行业研发运营一体化成熟度模型

DevOps maturity model for Internet application in insurance industry

（征求意见稿）

201X-XX-XX 发布　　XXXX - XX - XX 实施

中国保险行业协会　　发　布

目　　次

前　言

本标准按照GB/T 1.1-2009给出的规则起草

本标准由中国保险行业协会提出并归口

本标准起草单位：中国信息通信研究院，中国太平洋保险（集团）股份有限公司，中国人寿保险股份有限公司数据中心，中国人民财产保险股份有限公司，安心财产保险有限责任公司，中国再保险（集团）股份有限公司，阳光保险集团股份有限公司，华为技术有限公司，深圳市腾讯计算机系统有限公司，北京优帆科技有限公司，云栈科技（北京）有限公司，杭州数梦工场科技有限公司，北京易捷思达科技发展有限公司

本标准起草人：

引　言

研发运营一体化（DevOps）在软件的研发和交付过程中，将需求、开发、测试、部署和运营有效的统一，实现敏捷开发、持续交付和技术运营的集成。为了保证保险企业在构建时通过使用研发运营一体化，能够提高IT效能，在保证系统运行稳定的同时，快速交付高质量软件，本标准对保险行业研发运营一体化成熟度模型从敏捷开发、持续交付、技术运营、系统与工具四方面做出定义。

保险行业研发运营一体化成熟度模型

1 范围

本标准规定了保险行业研发运营一体化成熟度模型，包括敏捷开发、持续交付、技术运营、系统与工具四部分。

本标准适用于为保险行业云服务科技公司或保险业科技部门建设和实施研发运营一体化的过程中提供规范。

2 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。

GB/T 32400-2015 信息技术 云计算 概览与词汇

YD/T 1753-2018 研发运营一体化（DevOps）能力成熟度模型 第 1 部分：总体架构

YD/T 1754-2018 研发运营一体化（DevOps）能力成熟度模型 第 2 部分：敏捷开发管理

YD/T 1755-2018 研发运营一体化（DevOps）能力成熟度模型 第 3 部分：持续交付

YD/T 1756-2018 研发运营一体化（DevOps）能力成熟度模型 第 4 部分：技术运营管理

YD/T 1757-2018 研发运营一体化（DevOps）能力成熟度模型 第 5 部分：系统和工具

互联网保险业务监管办法（保监发〔2015〕69 号）

3 术语和定义

下列术语和定义适用于本文件。

3.1

部署流水线 deployment pipeline

指软件从版本控制库到用户手中这一过程的自动化表现形式。

[YD/T 1755-2018，定义3.4]

3.2

用户故事 user story

从用户的角度描述用户期望得到的功能。

[YD/T 1754-2018，定义3.1]

3.3

用户故事地图 user story mapping

将用户故事按一定顺序和优先级排列以分析与识别最小可行产品。

[YD/T 1754-2018，定义3.2]

3.4

配置项 configuration item

即纳入配置管理范畴的工作成果，是保存系统和项目的相关配置。

[YD/T 1755-2018，定义3.1]

4 面向保险行业的研发运营一体化流程

研发运营一体化（DevOps）过程见图1，主要包括以下流程：

——敏捷开发：随着保险行业新渠道、新业务的迅速推出和发展，敏捷开发IT架构对保险行业的销售、决策、管理等方面起着越来越大的作用。敏捷开发是一种应对快速变化的市场和技术环境的软件开发方法。强调价值交付过程中各类角色之间的紧密协作，主张演进式的规划和开发方式、持续和尽早的交付。

——持续交付：通过保险行业的项目流程管理、自动化的重复部署验证等手段来保证各项变更安全、快速、高质量地落实到生产环境或用户手中，缩短软件发布周期，降低交付风险。

——技术运营：保险行业应以业务为中心，交付稳定、安全、高效的技术运营服务。

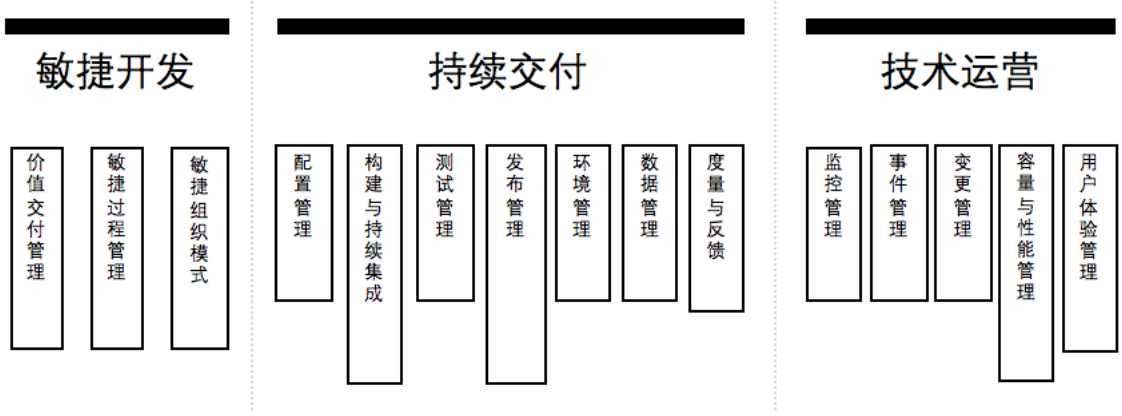

图1 研发运营一体化流程

5 面向保险行业的研发运营一体化成熟度等级划分

面向保险行业的研发运营一体化成熟度模型分为3个等级，包括基础级、增强级和先进级，如表1所示。每个级别按照不同程度说明，高级别内容宜包含低级别内容，无需重复引用。

表1 研发运营一体化成熟度等级划分

基础级	在企业内较大范围地推行DevOps并获得一定效率提升。
增强级	在企业内全面推行DevOps并在软件生命周期内获得整体效率提升。
先进级	在企业内全面落地DevOps并可达到整体效率最优化。

6 敏捷开发过程能力要求

6.1 价值交付管理

主要包括需求工件和需求活动两部分内容，体现需求管理过程中的分析、测试和验收三个阶段。

6.1.1 需求工件

对照表 2 给定的需求工件能力成熟度分级规则，确定需求工件能力成熟度级别。

表2 需求工件

	基础级	增强级	先进级
需求内容和形式	——进行需求分析并形成用户故事。 ——用户故事应满足:用户故事可协商和细化;规模可以在一次发布周期内完成;区分优先级。	同上一级 用户故事满足 INVEST 标准: ——独立完整性。 ——可协商和细化的。 ——有业务价值,能够进行价值评估。 ——能评估工作量和优先级。 ——足够小。 ——可测试。	同上一级 具有挖掘和分析需求价值的敏捷活动。
需求测试用例编写	建立测试用例与用户故事的关联,测试用例在需求分析结束、设计阶段完成。	同上一级	同上一级 测试和开发并行工作,形成测试用例。
需求测试用例验证	测试用例全部通过验证。	同上一级 使用工具自动执行部分测试用例。	同上一级
需求测试用例管理	测试用例无法重用。	能够对测试用例管理	同上一级 ——支持图形化的测试用例管理。 ——建立企业级可视化便捷的平台,管理包含测试用例的需求文档,可以通过需求文档查看产品的全貌。

6.1.2 需求活动

对照表 3 给定的需求活动能力成熟度分级规则，确定需求活动能力成熟度级别。

表3 需求活动

	基础级	增强级	先进级
需求分析	具有需求变更流程。	同上一级 团队中各个角色可共同对用户故事细化。	同上一级 具有改进需求分析协作的机制。
需求验收	——验收频率:每次交付都有验收。 ——验收范围:产品经理在每次交付时对交付成果进行验收。 ——反馈效率:能够把结果反馈给开发团队。	——验收频率:有稳定的交付，每次交付都有验收。 ——验收范围：产品经理、最终用户代表在每次交付时对交付成果进行验收。 ——反馈效率:能够把结果快速反馈给开发团队。	同上一级 ——验收范围:通过原型确认、AB 测试、灰度测试等方法进行验收测试。 ——反馈效率:能够快速响应用户反馈,建立企业级数据分析工具,分析用户行为数据。

6.2 敏捷过程管理

6.2.1 价值流

价值流是指产品经理、研发团队在软件研发过程中将软件产品转化为业务价值的能力，包括按照用户故事地图按需交付可用的软件，交付的软件能准确反映需求提出者的诉求，软件质量、用户体验能让使用者满意，软件研发过程中应具备将软件产品转化为业务价值的能力。对照表 4 给定的价值流能力成熟度分级规则，确定价值流能力成熟度级别。

表4 价值流

	基础级	增强级	先进级
交付	——产品经理、研发团队采用敏捷的方法提升交付价值。 ——约定软件质量指标。 ——有交付验收测试流程。	同上一级 ——具有稳定的交付节奏。 ——软件质量指标包括业务价值评估指标、业务准确性指标等。	同上一级 ——具有产品级回顾改进机制。
价值流	具有交付式管理模式。	同上一级 ——通过工具支撑计划安排活动，支持任务间和团队间的依赖管理。	同上一级 ——能够可视化交付速度等指标。

6.2.2 会议活动

会议活动能够可视化的管理价值流动，控制流动节奏，建立反馈机制，不断提升交付效率。对照表5给定的会议活动能力成熟度分级规则，确定会议活动能力成熟度级别。

表5 会议活动

	基础级	增强级	先进级
交付计划	针对需求分析、开发、测试、发布等不同阶段制定产品计划。	同上一级 ——团队围绕交付价值共同制定产品需求计划。	同上一级 ——能够灵活规划，不断改进。
交付活动	开展计划、评审会议，以快速有效的交付业务价值。	同上一级 ——具备措施减少变更带来的影响。	同上一级
人员组织	明确产品经理、敏捷教练、团队三类角色。	同上一级 ——建立特性团队。	同上一级 ——采用扁平化的敏捷团队组织架构。

6.3 敏捷组织模式

6.3.1 敏捷角色

敏捷角色应以价值交付为目标，持续提升交付效率。对照表6给定的敏捷角色能力成熟度分级规则，确定敏捷角色能力成熟度级别。

表6 敏捷角色

	基础级	增强级	先进级
敏捷角色	——不同角色具有明确分工。 ——每个角色具有专一的专业技术能力。 ——每个角色关注自身的工作。	同上一级 ——具有敏捷教练的角色。 ——每个角色在完成自己身工作的同时，能够快速变更角色。 ——团队能关注整体交付进度。	同上一级 ——没有敏捷教练的情况下团队依然能够有效运转。 ——团队成员能力趋于多样化，每个成员有强项，具备跨功能或角色的能力。 ——协作模式可形成借鉴或推广的经验积累。

6.3.2 团队结构

团队结构是以价值交付的最小实现单元构建最小化的功能团队。对照表7给定的团队结构能力成熟度分级规则，确定团队结构能力成熟度级别。

表7 团队结构

	基础级	增强级	先进级
团队结构	——团队足够小，10以下。 ——具有一致的约定。	同上一级 ——组建特性团队。	同上一级 ——能够持续提升团队。

7 持续交付过程能力要求

7.1 配置管理

7.1.1 版本控制管理

对照表 8 给定的版本控制管理能力成熟度分级规则，确定版本控制管理能力成熟度级别。

表8 版本控制管理

	基础级	增强级	先进级
版本控制	——具有版本控制系统。 ——支持分支管理。 ——使用制品库管理构建产物。	同上一级 ——将配置文件、构建和部署等自动化脚本纳入版本控制系统。 ——分支频繁地向主干合并。 ——所有交付制品纳入制品库管理。	同上一级 ——将软件生命周期的所有配置纳入版本控制系统管理。 ——持续优化的分支管理机制。 ——持续交付的制品管理机制。

7.1.2 配置变更管理

对照表 9 给定的配置变更管理能力成熟度分级规则，确定配置变更管理能力成熟度级别。

表9 配置变更管理

	基础级	增强级	先进级
变更管理	——记录代码变更信息。 ——对重点变更进行评审。 ——具有清晰的版本号规则。 ——手工回滚。	同上一级 ——所有配置项变更由变更系统触发。 ——每次变更都进行评审。 ——版本控制系统和变更管理系统自动化关联。 ——自动化回滚。	同上一级 ——可视化变更生命周期。 ——变更分级评审机制。 ——各个环节变更信息可追溯。

7.2 构建与持续集成

7.2.1 构建

构建指通过构建工具将软件代码转为可执行程序的过程。对照表10给定的构建能力成熟度分级规则，确定能力成熟度级别。

表10 构建

	基础级	增强级	先进级
构建	——通过脚本自动化构建。	同上一级	同上一级

	——有独立的构建服务器。 ——每日自动构建。 ——构建环境和工具由专人负责维护。	——结构化的构建脚本。 ——构建环境配置标准化，有独立的构建资源池。 ——定期自动构建，明确构建计划和规则。 ——构建环境和工具由细分的团队人员负责维护。	——构建方式服务化。 ——构建资源动态弹性按需分配与回收。 ——按需制定构建计划。 ——构建能力赋予全部团队成员。

7.2.2 持续集成

持续集成是软件工程领域中的一种最佳实践，即鼓励研发人员频繁的向主干分支提交代码，频率为至少每天一次。每次提交都触发完整的编译构建和自动化测试流程，缩短反馈周期，及时修复问题，从而保证软件代码质量，减少大规模代码合并的冲突和问题，软件可按照指定时间发布。对照表11给定的持续集成能力成熟度分级规则，确定持续集成能力成熟度级别。

表11 持续集成

	基础级	增强级	先进级
持续集成	——统一的持续集成服务。 ——几天或几周集成一次。 ——代码集成作为软件交付中的一个独立阶段。 ——集成问题反馈和解决周期以天计算。	同上一级 ——专门的持续集成团队。 ——至少每天集成一次。 ——集成问题反馈和解决在几个小时内完成。	同上一级 ——持续优化和改进团队持续集成服务。 ——每天多次集成的能力。 ——集成问题反馈和解决在半个小时内完成。

7.3 测试管理

7.3.1 测试分层策略

对照表 12 给定的测试分层策略能力成熟度分级规则，确定测试分层策略能力成熟度级别。

表12 测试分层策略

	基础级	增强级	先进级
测试分层策略：	——已建立分层策略。	同上一级 ——测试设计以对接口/服务级测试为主。	同上一级 ——测试设计以对代码级测试为主。

7.3.2 代码质量管理

在代码变更后，应对代码质量进行检查、分析，并针对问题给出改进建议。对照表 13 给定的代码质量管理能力成熟度分级规则，确定代码质量管理能力成熟度级别。

表13 代码质量管理

	基础级	增强级	先进级
代码质量管理	——已建立团队级代码质量规约。覆盖部分代码质量指标，如代码规范、错误复杂度等。 ——采用自动化结合手工方式进行代码质量检查。 ——对代码质量检查结果给出反馈，只处理部分检查结果。	——已建立组织级代码质量规约。将安全漏洞检查、合规检查纳入规约。 ——采用完全自动化的方式进行代码质量检查。 ——对代码质量检查结果及时处理。	——建立公司级代码质量规约。定期对规约进行优化。 ——具备企业级代码质量管理平台，以服务的形式提供对代码质量的检查分析。 ——对代码质量数据进行统一管理，可有效追溯代码质量。

7.3.3 自动化测试

对照表 14 给定的自动化测试能力成熟度分级规则，确定自动化测试能力成熟度级别。

表14 自动化测试

	基础级	增强级	先进级
自动化测试	——对业务级的 UI 测试进行自动化设计。 ——专人统一管理自动化测试脚本与工具。 ——支持自动化执行。 ——具备一定的自动化分析能力。	——对接口/服务和代码级测试进行自动化设计。 ——具有统一的自动化测试框架。 ——自动化测试由流水线自动化触发。 ——具有较强的自动分析能力。	——对性能、稳定性、安全性等非功能性测试进行自动化测试。 ——建立自动化测试自服务平台。 ——定期验证自动化执行策略并持续优化。 ——对自动化测试结果智能分析。

7.4 发布管理

7.4.1 部署模式

对照表 15 给定的部署模式能力成熟度分级规则，确定部署模式能力成熟度级别。

表15 部署模式

	基础级	增强级	先进级
部署模式	——运维人员通过自动化脚本实现部署。 ——流程文档标准化。 ——以周为单位定期部署。 ——部署失败率中等。	——部署和发布全自动化。 ——使用相同的过程和工具完成所有环境部署。 ——以天为单位定期部署。 ——部署失败率中低。	——持续化的部署发布模式和工具系统平台。 ——每次变更都触发自动化部署。 ——可进行安全可靠地部署与发布。 ——具有持续监控体系，出现问题自动回滚。

7.4.2 部署流水线

部署流水线应将复杂的交付流程分割为多个阶段，每个阶段层层递进，快速反馈。对照表 16 给定的部署流水线能力成熟度分级规则，确定部署流水线能力成熟度级别。

表16 部署流水线

	基础级	增强级	先进级
部署流水线	——具有完整的交付过程和规范。 ——交付环节自动化。 ——交付过程可追溯。	同上一级 ——交付仅在必要环节进行手工确认。 ——团队内共享度量指标。	同上一级 ——团队间依赖解耦，可独立完全的自主部署交付。 ——持续部署流水线驱动持续改进。 ——部署流水线信息可进行数据价值挖掘。

7.5 环境管理

环境管理以最小的代价确保一致性。对照表 17 给定的环境管理能力成熟度分级规则，确定环境管理能力成熟度级别。

表17 环境管理

	基础级	增强级	先进级
环境管理	——建立生产环境、功能测试环境。 ——环境构建通过自动化来完成，准备时间以天为单位。 ——通过配置管理工具实现操作系统级别的依赖管理。	同上一级 ——标准的研发环境。 ——环境构建通过自服务的资源交付平台来完成，环境准备时间以小时为单位。 ——有服务级依赖的管理配置能力。	同上一级 ——建立全面的测试与灰度环境。 ——环境构建可以通过容器化快速交付，环境准备时间以分钟级为单位。 ——环境和依赖配置管理实现代码化描述，可以做到实例级的动态配置管理能力，根据业务和应用架构弹性变化。

7.6 测试数据管理

对照表 18 给定的测试数据管理能力成熟度分级规则，确定测试数据管理能力成熟度级别。

表18 测试数据管理

	基础级	增强级	先进级
数据管理	——导出部分生产环境数	同上一级	同上一级

	据形成基准的测试数据集。 ——测试数据覆盖正常类型、错误类型、边界类型等。 ——测试数据具有明确的备份恢复机制。	——对从生产环境导出的数据进行漂白。 ——覆盖全部测试分层策略要求的测试类型。 ——测试用例的执行不依赖其他测试用例执行所产生的数据。	——所有数据可通过模拟、调用 API 的方式自动生成。 ——持续优化的持续数据管理方式和策略。 ——对测试数据分级。

7.7 度量与反馈

对照表 19 给定的度量与反馈能力成熟度分级规则，确定度量与反馈能力成熟度级别。

表19 度量与反馈

	基础级	增强级	先进级
度量指标	——持续交付的各个阶段定义度量指标。 ——度量指标以结果指标为主。 ——度量数据采用抽样方法收集。	同上一级 ——建立跨组织的度量指标。 ——度量指标覆盖过程指标。 ——持续收集度量数据。 ——度量指标按需求定期更新。	同上一级 ——持续优化的度量指标。 ——度量指标覆盖探索性指标。 ——对历史度量数据进行数据分析。 ——度量指标可基于大数据分析和人工智能自动识别和推荐动态调整指标优先级。

8 技术运营过程能力要求

8.1 监控管理

能够对研发运营过程中的对象进行数据采集、处理、分析、异常识别与通知等操作。

8.1.1 指标采集

对照表20给定的指纹采集能力成熟度分级规则，确定指纹采集能力成熟度级别。

表20 指纹采集

	基础级	增强级	先进级
指标采集	——支持对主机、网络、中间件、业务应用的监控。 ——具有完善的主动采集插件和任务框架。 ——业务应用的监控数据误差小于 1%。	同上一级	同上一级

	——具备秒级上报的实时性。		

8.1.2 监控数据处理

对照表21给定的监控数据处理能力成熟度分级规则，确定监控数据处理能力成熟度级别。

表21 监控数据处理

	基础级	增强级	先进级
监控数据处理	——在单机上部署少量程序对小量的数据加工处理。 ——对监控数据建立关系模型存储。 ——应用于特定领域的监控场景。	同上一级 ——在中小型集群上部署数据处理程序。 ——抽象监控数据模型。 ——应用于复杂的领域监控场景。	同上一级 ——在通用的分布式流处理集群对多种类型的海量数据进行加工处理。 ——抽象多维数据模型。 ——应用于复杂领域的精细化监控场景。

8.1.3 异常识别

对照表22给定的异常识别能力成熟度分级规则，确定异常识别能力成熟度级别。

表22 异常识别

	基础级	增强级	先进级
异常识别	——通过阈值识别异常点。 ——对异常事件按时间、告警对象等维度进行告警合并。 ——告警延迟在 3 分钟内。	同上一级 ——集中聚合多个待检测指标进行异常识别。	同上一级 ——采用指标分级方法对重点指标进行异常识别。

8.1.4 监控可视化和通知

对照表23给定的监控可视化和通知能力成熟度分级规则，确定监控可视化和通知能力成熟度级别。

表23 监控可视化和通知

	基础级	增强级	先进级
监控可视化及通知	——短期内恢复的异常，能够自动消除告警。 ——短信、邮件等方式通知告警。 ——能够按照告警项、级别、时间等维度生成报表。	同上一级 ——有自动化脚本收集告警信息。	同上一级 ——能够展示调用链各个环节的异常情况。 ——有预处理脚本或工具，能针对告警、性能进行处理，处理完毕后能自动消除

			告警 ——能够自动生成报表。

8.2 事件管理

8.2.1 事件发现

对照表 24 给定的事件发现能力成熟度分级规则，确定事件发现能力成熟度级别。

表24 事件发现

	基础级	增强级	先进级
事件发现	——建立统一的服务台受理事件。 ——根据影响度和紧急度划分事件优先级。 ——建立基本的工具记录事件。	同上一级 ——对不同级别和类别的事件设定对应的服务级别。 ——建立用户自助 IT 服务门户。	同上一级 ——分析各类事件的发生趋势。 ——根据事件处理中影响度和紧急度的变化，自动调整事件级别。 ——用户完全感知不到事件发生，而且 IT 主动引领业务创新。

8.2.2 事件处理

对照表 25 给定的事件处理能力成熟度分级规则，确定事件处理能力成熟度级别。

表25 事件处理

	基础级	增强级	先进级
事件处理	——统一定义各等级事件服务级别。 ——设定一、二、三线运维支持团队。	同上一级 ——重大事件应急处置机制运行顺畅。 ——明确各团队的 KPI 考核指标。	同上一级 ——大部分事件可自动修复。

8.2.3 事件回顾

对照表 26 给定的事件回顾能力成熟度分级规则，确定事件回顾能力成熟度级别。

表26 事件回顾

	基础级	增强级	先进级
事件回顾	——将重大、典型或重复故障生成问题进行处理，并形成知识。	同上一级 ——建立专业的知识录入和分享工具。	同上一级 ——形成的知识库，能录入自动化工具，实现故障自动解决恢复。。

8.3 变更管理

对照表 27 给定的变更管理能力成熟度分级规则，确定变更管理能力成熟度级别。

表27 变更管理

	基础级	增强级	先进级
变更管理流程	——具有规范的变更管理流程。	同上一级	同上一级
变更管理人员	——设立变更管理岗位。	——需要运维现场值守。	——运维只需要远程值守。
变更管理工具	——有变更管理系统。	——变更管理系统支持多种灰度模式自动变更。	——全面的自动变更管理系统。
变更指标	——变更失败率小于 3%。	——变更失败率小于 1%。	——失败变更率小于 0.5%。

8.4 容量和性能管理

对照表 28 给定的容量和性能管理能力成熟度分级规则，确定容量和性能管理能力成熟度级别。

表28 容量和性能管理

	基础级	增强级	先进级
容量管理	——有基本的容量管理活动。 ——有监控和测试工具。	——有明确的容量管理制度。 ——容量管理的每个阶段都有工具支撑。	——有明确的容量管理制度和容量指标识别分析。 ——对每个阶段的数据进行分析。
性能管理	——对基础性能指标建立起有效的管理控制能力。	——对应用服务、架构、用户体验等有性能度量。	——端到端的性能管理能力，能够进行趋势分析。

8.5 成本管理

对照表 29 给定的成本管理能力成熟度分级规则，确定成本管理能力成熟度级别。

表29 成本管理

	基础级	增强级	先进级
决策机构	——理顺流程,有合理性依据。 ——颁布预算管理流程规范。 ——分析总成本、现金流和产品 KPI 增长是否吻合。	同上一级 ——建立合理的资源量推导模型和产品指标的预测，增加预算滚动机制。 ——不同部门、产品预算执行率分析。	同上一级 ——人工和机器学习同时优化资源模型,带宽预算使用正向推导,而不是历史数据反推。
成本优化	——建立应用资源使用标准规范。 ——引入中间件层，应用层,数据层支持有限水平扩展能力。	同上一级 ——打通持续交付流程,用户自助完成测试、生产资源申请及代码发布。	同上一级 ——进行动态编排,峰值自动扩容，峰谷自动缩容。 ——按需交付,自动化工具部署。

8.6 连续性和可用性服务

对照表 30 给定的连续性和可用性服务能力成熟度分级规则，确定连续性和可用性服务能力成熟度级别。

表30 连续性和可用性服务

	基础级	增强级	先进级
连续性管理	——有 IT 连续性管理流程。 ——有 IT 灾难恢复计划。 ——完全数据备份至少每月一次，数据同城备份 ——不定期的桌面检查、走查、模拟演练。	同上一级 ——有业务连续计划。 ——数据异地备份。	同上一级 ——有业务连续性管理。 ——数据多于 2 份备份 ——定期桌面演练、沙盘演练、模拟演练、部分系统演习、全面演习
可用性管理	——同城灾备中心。 ——部分关键 IT 系统高可用架构设计。 ——实施简单的主动的可用性管理 ——持续集成失败平均故障修复时间<8 小时;	同上一级 ——异地灾备中心。 ——个别关键 IT 系统容错架构设计。 ——主动的可用性趋势分析，采取主动措施。 ——持续集成失败平均故障修复时间<4 小时	同上一级 ——分布式多活数据中心。 ——全部 IT 系统高可用架构设计；大部分关键 IT 系统容错设计。 ——续集成失败平均故障修复时间<2 小时。
应用事件管理	——用户先于维护人员发现事件，有事件响应团队。 ——有通知和报告的自动化工具。	同上一级 ——拥有应急预案优先恢复业务。 ——协同工具进行事件会诊处理	同上一级 ——事件能事先自动预警，有事件管理规范并执行，有各种场景的应急预案。 ——有自动预警的手段，有事件处理过程的自动化工具。

8.7 用户体验管理

8.7.1 业务认知

对照表 31 给定的业务认知能力成熟度分级规则，确定业务认知能力成熟度级别。

表31 业务认知

	基础级	增强级	先进级
业务认知	——了解业务流程。 ——有基本的培训。	同上一级 ——掌握核心业务流程。	同上一级 ——精通核心业务流程。

8.7.2 体验优化

对照表 32 给定的体验优化能力成熟度分级规则，确定体验优化能力成熟度级别。

表32 体验优化

	基础级	增强级	先进级
体验优化	——通过日志发现体验日常。 ——支持异常体验定位。 ——手工修复体验异常。	——主动监控发现体验异常。 ——支持复杂环境下的异常体验定位。 ——有体验异常决策能力。 ——标准化工具修复体验异常。	——端到端的性能管理能力，能够进行趋势分析。

9 系统和工具能力要求

9.1 项目管理

9.1.1 需求与任务管理平台

平台对项目设计与开发过程中所有需求、计划和任务进行管理。应包含以下基本功能：

——支持优先级设置，不同优先级表示不同严重级别或重要程度；

——支持状态设置与变更，不同状态表明需求、计划和任务所处的阶段。如任务创建于 Open 状态，然后开始执行/Progress，再到完成/Finished，最后被关闭/Closed。根据情况的不同，用户可以根据项目来定制状态以及工作流；

——支持分类管理和关键字标识；

——支持可视化面板，可以简单地创建、复制，生成多个面板，面板可以展示项目统计报表。

——至少支持邮件、RSS、即时通讯中的一种通知方式，能在项目关键阶段自动发送通知。支持非项目参与人（具有项目权限）关注项目动态并接收到通知。系统页面明显的位置发布最新通知公告；

——安全与权限。应指定项目负责人，支持项目指派、再指派、认领、二次认领。可以自定义安全级别，不同用户对项目有不同权限；

——支持项目关联；

——支持项目处理流程跟踪；

——支持搜索。

9.1.2 文档与知识管理平台

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。应包含以下基本功能：

——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、关键字等快速查找到所需的文档；

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、技术分享等。应包含以下基本功能：

——支持词条创建、编辑、分类和分享；

——支持多人协同编辑；

——支持搜索；

——可导出为常见文档格式（例如Word、PDF、PPT等）；

——支持上传常见文档格式的附件（例如Word、PDF、PPT等）；

——宜支持Wiki、Markdown格式。

9.1.3 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：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：需求从提交到交付的时长；

——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：代码从提交到交付的时长；

——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。

9.2 开发管理

9.2.1 代码管理平台

应包含以下基本功能：

——版本仓库：支持版本仓库的建立、删除、分类、复制、派生、限额、扩容、共享与可见范围；

——分支管理：支持分支的创建、删除、追溯、分类和识别；

——权限管理：支持权限的分级，如查看、提交、合并主干等权限；

——变更与合并管理：支持变更的追溯和回滚，支持合并的追溯；

——基线管理：支持基线的创建、删除、追溯、分类和识别；

——代码Review：支持Review的发起和管理；

——存储和备份：支持支持代码的存储和备份；

——安全保障：账号具有唯一性；支持重要操作保护；支持日志审计与回溯。

9.2.2 代码质量管理

代码质量管理提倡用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，提升开发效率，节省开发成本。应该包含以下基本功能：

——支持Java、C/C++、JavaScript、PHP等多种主流编程语言的代码质量检测；

——持续检查，能够提供代码缺陷概览，并实时跟踪新增代码引入的缺陷、已有缺陷修复情况；

——集成到整个DevOps工具链中，定时/实时自动化开展；

——代码质量符合度标准应不断审视和优化，能够看到项目的持续改进；

——项目核心代码工具检查覆盖率为100%；

——代码质量报告能够自动生成，要求有新增缺陷，修复缺陷，遗留缺陷以及相关趋势等量化质量指标；

——能够看到项目组制定的缺陷修复计划和行为，遗留缺陷趋势保持收敛下降。

9.3 集成与部署管理

9.3.1 持续集成

应包含以下基本功能：

——保存多个构建项目；

——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；

——构建项目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。

9.3.2 制品管理

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，应用配置文件等。应包含以下基本功能：

——支持npm、bower、rpm等更多种类的制品类型；

——为制品添加元数据信息；

——使用制品的审计日志；

——基本的权限管理；

——检索制品；

——备份和恢复。

9.3.3 部署管理

应包含以下基本功能：

——自动打包；

——支持编排部署步骤，可以根据业务场景自定义部署流程；

——可视化：仪表盘支持显示部署活动状态等内容；

——支持Docker等多种部署运行方式；

——支持部署活动审计，日志信息可发送到日志分析系统；

——API接口：支持应用系统调用部署系统能力进行部署。

9.3.4 发布管理

发布管理是将通过构建的程序，发布到软件环境中。应包含以下基本功能：

——发布规划，规划软件程序的整体发布计划，包含但不限于：发布窗口、发布策略、发布执行、发布确认，以及发布风险的预估；

——发布窗口，程序发布的具体日期时间；

——发布策略，通过选择进行发布的实例、发布并发度、超时时间、暂停点、软件版本等发布的具体策略，执行对应的发布动作；发布策略包含并不限于原地发布、金丝雀发布、蓝绿发布等；
——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。

9.3.5 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；
——可以根据环境的配置快速生成交付环境；
——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；
——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；
——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。

9.4 测试管理

9.4.1 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；
——设置用例集、子用例集的名称、标签、状态；
——设置用例名称、描述、标签、状态、优先级、是否自动化、设计人员；
——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；
——设置用例集、子用例集、用例和需求、特性、故事的关联；
——设置自动化测试用例和测试脚本的关联。

9.4.2 缺陷管理

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：

——描述缺陷内容，支持上传视频、图片等附件；
——标记缺陷优先级；
——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；
——可添加评论；
——指派缺陷、更改缺陷状态，发送消息给相关人员；

——可按照指派人、优先级、当前状态等维度过滤缺陷；
——更新状态的操作有权限控制，缺陷要进入不同的状态需要特定角色或特定人员才能操作；
——区分缺陷的解决状态和关闭状态，开发人员标记为已解决的缺陷，被验证后再关闭；
——缺陷可关联到修复该缺陷的代码。

9.4.3 测试数据管理

测试数据管理是指在测试过程中完成数据收集、生成、维护、自动化的过程。应包含以下基本功能：
——数据仓库，支持用户存储、扩充、共享和重用测试数据集，以提高测试效率；
——测试数据生成：当生产数据不能直接用来进行测试时，为测试提供按需创建的生产质量数据，允许测试人员根据业务规则和限制条件快速创建复杂的数据集；
——支持敏感数据发现与脱敏。通过降低数据敏感性、匿名化敏感数据、对数据进行假名处理等手段，创建可在内外部安全共享的真实匿名化数据，避免敏感数据泄露；
——支持在多应用系统和数据库中创建和管理数据子集，减少测试数据占用的空间和存储时间。

9.4.4 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。

9.4.5 性能测试

应包含以下基本功能：
——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。

9.5 技术运营管理

9.5.1 CMDB

配置管理数据库（CMDB）存储与管理企业IT架构中设备的配置信息。应包含以下基本功能：
——可视化管理：能够可视化展示拓扑信息、资源数量、资源使用情况、资源变化趋势等内容；
——数据的导入和导出：支持通过Excel等形式导入配置信息，提供表格导出；
——建立自定义CI来定义及管理需要的对象；
——属性自定义：用户可以选择需要展示的属性字段，也可以增加平台没有的属性字段；
——分类管理：可从业务、集群等多种维度进行分类管理；
——操作审计：用户操作记录可追溯；
——资源自动发现；
——分钟级配置数据一致性校验；
——API接口：通过接口保证CMDB数据的一致性。

9.5.2 作业平台

应包含以下基本功能：
——脚本管理：支持脚本的新建、自动执行、编辑和删除，可以通过手动编写、上传、已有脚本克隆等方式导入脚本；
——支持大文件拉取/分发，支持本地上传和服务器上传两种方式；
——支持多个脚本或文件分发的节点串接组合后执行；
——常用作业执行，可对已保存作业任务进行“执行、克隆、编辑、定时、删除”等操作；
——支持高并发执行任务；
——支持秒级定时任务；
——API接口：提供API接口供其他系统或平台调度；
——操作审计：能够对脚本执行、文件分发、API调用、定时任务等操作进行记录和追溯。

9.5.3 监控管理

应包含以下基本功能：
——指标收集：支持服务器、虚拟机、网络设备等多种设备，保险业务系统、内外部接口（如保险承保、支付确认）等多种系统和应用程序的监控，能够自动完成指标采集；
——问题检测：支持自定义阈值、策略，自动检测采集指标的问题状态；
——可视化管理：通过仪表盘、网络图、表格等形式呈现监控对象环境状态；
——自动发现：能够主动代理、自动注册；
——通知：至少支持邮件、RSS、即时通讯中的一种通知方式，能在出现问题时及时发送通知。

9.5.4 日志分析

应包含以下基本功能：

——日志采集与存储：支持采集服务器、网络设备、保险业务系统、内外部接口的日志，留存取证；

——快速搜索查询：支持海量日志快速和多维度查询，包括范围查询、正则表达式、模糊匹配等方式。能够对查询字段进行定位日志上下文；

——核心业务统计分析：支持多种统计分级函数。能够对理赔、续保等保险核心系统的业务进行汇总和分析，生成业务统计报表，支持通过接口调用，作为决策依据，为问题排查提供参考信息；

——用户画像：掌握用户特征，能够对用户偏好做出反应和判断；

——可视化报表。