GOPS2017
Shanghai

GOPS

GOPS 全球运维大会 2017 上海站

指导单位：

主办单位：

大会时间：2017年11月17日-18日

大会地点：上海光大会展中心国际大酒店（上海徐汇区漕宝路67号）

DevOps 标准解读（Beta版）

栗蔚

关于DevOps，那些常见的只言片语

- DevOps 是自动化运维
- DevOps 是运维会开发
- DevOps 用容器实现工具
- DevOps 是敏捷开发
- DevOps 玩玩沙盘
- DevOps 是一种最佳实践，没法统一规则

DevOps 标准，终结盲人摸象的时代

“研发运营一体化能力成熟度模型”，是国内外第一个 DevOps 标准体系。

- 三正
  - 正概念
  - 正框架
  - 正能力
- 三明
  - 明流程
  - 明组织
  - 明实施

DevOps 系列标准（Beta版）

GOPS2017
Shanghai

- 研发运营一体化能力成熟度模型 第1部分：总体架构（已完成）
- 研发运营一体化能力成熟度模型 第2部分：敏捷开发过程（已完成）
- 研发运营一体化能力成熟度模型 第3部分：持续交付过程（已完成）
- 研发运营一体化能力成熟度模型 第4部分：技术运营过程（制订中）
- 研发运营一体化能力成熟度模型 第5部分：应用架构（制订中）
- 研发运营一体化能力成熟度模型 第6部分：组织结构（制订中）

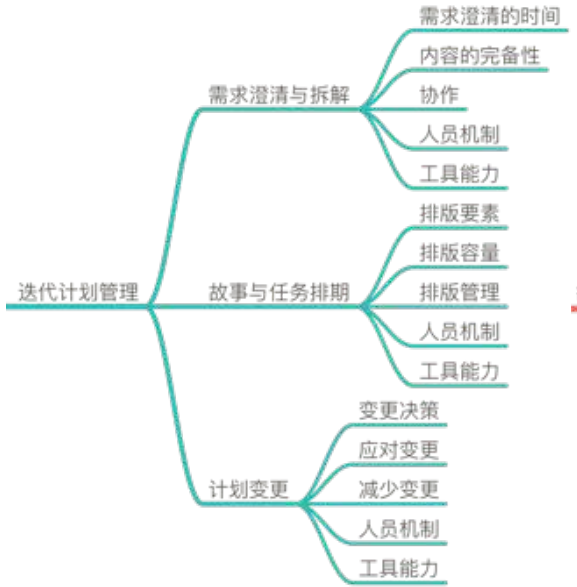
研发运营一体化能力成熟度模型 第1部分：总体架构

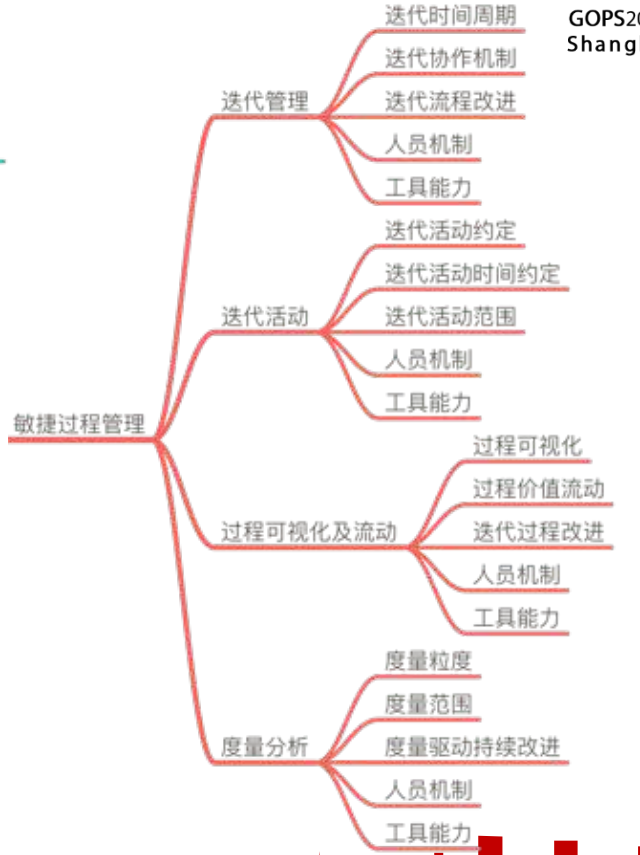
GOPS2017 Shanghai

一、研发运营一体化（DevOps）过程															
敏捷开发管理			持续交付								技术运营				
需求管理	计划管理	过程管理	配置管理	构建与持续集成	测试管理	部署与发布管理	环境管理	数据管理	度量与反馈	安全管理与服务	监控服务	数据服务	容量服务	连续性服务	运营反馈
需求收集	需求澄清和拆解	迭代管理	版本控制	构建实践	测试分级策略	部署与发布模式	环境供给方式	测试数据管理	度量指标	安全开发	应用监控	数据收集能力	容量规划能力	高可用规划	业务知识管理
需求分析	故事与任务排期	迭代活动	版本可追踪性	持续集成	代码质量管理	持续部署流水线	环境一致性	数据变更管理	度量驱动改进	运营安全	质量体系管理	数据处理能力	容量平台服务	质量体系管理	项目管理
需求与用例	计划变更	过程可视化及流动			测试自动化					数据安全	事件响应及处置	数据告警能力	运营成本管理		业务连续性管理
需求验收		度量分析								风险与威胁模型	监控平台				运营服务管理
二、研发运营一体化（DevOps）应用架构															
三、研发运营一体化（DevOps）组织结构															

级别	英文	中文
1级	Regressive	阻碍的
2级	Repeatable	可重复的
3级	Consistent	一致的
4级	Quantitative	量化的
5级	Optimizing	优化的

研发运营一体化能力成熟度模型 第2部分：敏捷开发过程





敏捷开发过程-需求管理-需求收集

需求收集环节是需求提出方和产品经理之间明确产品需求的阶段，是产品研发运营一体化最初始阶段，把产品的需求具象化，形成待办事项列表的过程。

需求收集环节包括三个方面工作：

1）明确单个需求点：即以问题驱动为核心，探索问题核心相关事项的过程；

2）梳理需求全貌：应能列出为了落实产品的愿景而需要完成的所有事项，即待办事项列表；

3）确定待办事项列表：应包括用户需求所涉及的所有事项，并且作为产品研发路线图。

级别	主要工作完备性			人员机制	工具能力	备注
	单个需求点	需求全貌	需求的管理			
1	应能明确需求问题，制定明确的功能点需求要求	应梳理所有需求问题，形成需求说明书，涵盖所有的需求功能要求。	应有模板和规范，并在形成需求说明书之后的需求沟通、实施过程中应采用契约的方式传递。	无	无	
2	应通过协作方式形成适当详细的需求说明。	应有待办事项列表来管理需求。	需求应在需求进入迭代开发之前可以进行变更和细化。	需求提出方和产品经理应有明确的需求收集流程，制定了快速沟通协作机制，例如明确规定计划和需求之间的流转和协作方法和规范。	无	
3	同上	同上	同上且产品经理对提出的需求在产品演进过程中持续细化和演进，形成产品路线图。例如，采用精益产品的方法、影响力地图、MVP 的方法等敏捷方法等。	同上	需求提出方和产品经理应通过需求收集可视化工具，归集到待办事项列表，由产品经理统一管理。	
4	同上	同上	同上	同上且需求提出方和产品经理之间的机制应不限定时间和角色，保证需求随时入和出。例如，建立运营驱动需求的体系，在产品演进过程中，不断涌现需求，能不断优化和调整待办列表的顺序。	同上且有协作型需求沟通工具，在需求提出、收集、分析、实施过程中，各角色随时沟通都能对需求内容进行持续演进和细化。	
5	同上	同上	同上且建立需求快速上线、快速反馈流程，用户反馈能快速收集。	同上且企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人力、绩	同上且使用企业提供的统一的需求收集工具、协作型需求沟通工具，归集到待办事项列表，由产品经理	

敏捷开发过程-需求管理-需求分析

需求分析是产品经理将需求细化和拆解成用户故事的过程，主要体现三个方面：

1）明确需求内容和形式：需求分析形成用户故事，用户故事描述用户场景；

2）需求分析协作：用户故事是适度详细并适应变化的，可以在开发过程中对其进行评估不断细化；

3）需求管理方式：用户故事统一管理，并按照业务价值由高到低排定优先级。

级别	主要工作完备性			人员机制	工具能力	备注
	需求内容和形式	需求协作	需求的管理			
1	需求分析形成软件需求规格说明书，作为需求提出方和实施方之间的契约	需求分析人员在完成需求规格说明书的编写后离场，开发团队按照需求规格说明书进行开发。	通过需求规格说明书统一管理。	无	无	
2	需求分析形成用户故事，用户故事规模适中，可在一个迭代内完成。	迭代开始前，由产品经理、需求提出方开发团队一起细化用户故事。	应使用产品待办列表和迭代待办列表管理。	无	无	
3	用户故事符合INVEST标准：1）故事是独立完整的，2）故事是可协商并细化的，3）故事是有业务价值的，4）故事是能评估工作量和优先级的，5）故事是足够小的，一般在1-2日内完成，6）故事是可测试的。	在软件过程的任何阶段，产品经理、需求提出方及团队成员可对用户故事进行变更和细化。	同上且当发生规模型产品研发情况，应建立跨团队的产品待办列表，迭代待办列表	无	无	
4	同上且应具备可视化的MVP的产品演进路线，管理用户故事和发布迭代关系，可以使用例如：用户故事地图、影响力地图等敏捷方法。	同上且当发生跨团队的产品研发情况，应建立史诗故事、特性故事、用户故事的分层管理，可跨团队进行需求拆解细化。	同上且产品待办列表应符合DEEP原则：1）适当的详细描述的，优先级越高越详细明确，2）用故事点进行估算过大小的，3）随着产品演进不断涌现和变化的，4）优先级从高到低排序的。	应建立特性型研发团队，与产品经理合作提升需求分析落地的价值流动。	有协作型用户故事沟通工具、产品待办列表管理工具。	
5	同上	同上	同上且应建立需求与企业级活动关联，把企业战略和目	同上且企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品	同上且应建立企业级的需求管理工具。	

敏捷开发过程-需求管理-需求与用例管理

需求与用例管理是指产品经理和开发团队把用户故事的验收标准和测试用例进行关联性，能验收产品功能是否满足用户故事的要求的过程。主要体现在三个方面：

1）梳理需求用例：编写需求验收标准，形成测试用例的过程；

2）使用需求用例：需求用例指导需求开发，验证产品功能的过程；

3）管理需求用例：建立需求与用例的统一管理库，持续的使用和优化。

级别	主要工作完备性			人员机制	工具能力	备注
	需求与用例编写	需求用例验证	需求与用例的管理			
1	测试用例与需求没有关联，测试用例在设计结束代码开发阶段完成。	无	测试用例在本需求功能测试完成后没有做归档重用，在每次有新需求重新设计测试用例。	无	无	
2	测试用例与用户故事应有关联，测试用例在需求分析结束设计阶段完成。	每次上线前应把编写的测试用例全部验证通过，才可上线。	需求文档和测试用例应作为知识沉淀下来，当设计现有产品进行功能优化的需求时，需求文档和测试用例在现有的知识上进行调整优化。	无	无	
3	同上	同上	测试用例应作为产品的软件代码资产存在，所有的功能上线都以能测试用例验证通过为目标，每次迭代上线都必须执行产品沉淀下的所有测试用例，直到验证和修复通过才可上线。	无	测试用例能通过工具自动执行。	
4	同上且产品的需求在最初始阶段即转化为测试用例，细化需求编写验收标准过程即编写测试用例的过程。	同上	同上且需求作为需求用例库作为产品的软件代码资产存在，既保持可读性又作为用例在产品迭代更新中一直保持完整和准确。 同上且所有的功能上线都以能被可读的需求用例验证通过为目标，每次迭代上线都必须执行沉淀下的所有的需求用例，直到验证和修复通过才可上线。 当产品进行升级重构时，产品的需求用例库已经重建就能作为升级重构后的验收标准。	无	同上	

敏捷开发过程-需求管理-需求验收

需求验收是指产品经理、需求提出者和最终用户对产品的功能验收，要求能对需求进行快速测试、快速确认、快速反馈、快速优化。本节的需求验收，仅是指功能验收，非功能测试不在本节的范围内。需求验收主要体现在以下三个方面：

1）需求验收的频率：指不同角色对需求功能验收的频率，频率越高效果越好；

2）需求验收的范围：指需求验收应尽量具备有业务价值的端到端的验收；

3）需求验收的反馈效率：指需求验收的结果能准确、快速的反馈到开发团队的过程。

级别	主要工作完备性			人员机制	工具能力	备注
	需求验收频率	需求验收范围	需求验收反馈效率			
1	在项目末尾，需求上线后，一次性的实施alpha测试、beta测试、正式验收测试。	需求提出者或最终用户应对全量功能进行验收。	有验收测试流程，能把结果反馈到产品经理和开发团队。	无	无	
2	在每个敏捷迭代，应有验收评审会。	在验收评审会上，产品经理应对团队的迭代成果进行验收。	同上	无	无	
3	同上且在跨团队产品里，有跨团队的产品验收会，并要求在每个迭代都须召开。	同上且需求提出者或最终用户应能在每个发布后进行验收。	对验收测试应有快速的反馈和优化流程，能保障反馈能在进入产品待办列表，且根据优先级进入迭代待办列表。	验收须有产品经理、需求提出者和最终用户等参与。	无	
4	同上	同上且在迭代过程中，应有通过原型确认、AB测试、灰度测试等方法进行验收测试，提升验收效果	同上且建立产品级的业务价值验收反馈流程，在产品推向市场后，能在1-2个迭代就能快速进行响应。	同上	应有快速的反馈和优化流程和工具，能收集验收结果，并且能快速转化为迭代需求。	
5	同上	同上	同上且针对反馈的情况，能通过反馈发现迭代中的沟通、设计等各类问题，并进行持续改进。	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人力、绩效、核算等。敏捷团队以业务价值为核心以运营为驱动的敏捷工作模式，企业为团队提供IT基础设施、基础管理等支持。	应建立企业级大数据分析工具，能抓取用户行为数据，通过大数据分析，在用户功能验收和用户体验时作为辅助决策依据，持续优化改进。	

敏捷开发过程-迭代计划管理-需求澄清

需求澄清是产品经理和开发团队沟通和确认需求的过程，包含沟通和明确用户故事的细节（包括但不限于背景信息、UI和交互设计、测试要点等），确定用户故事的技术实现方案，识别技术风险和依赖，团队对用户故事进行任务拆分，产品经理和团队对于以上信息达成共识，明确用户故事完成的定义。

1）需求澄清的时间：指需求澄清发生在研发过程中的合适的阶段,以便适应研发过程中的变化及开发团队工作的开展。

2）需求澄清内容的完备性：指在需求澄清过程中，是否澄清需求的所有内容。

3）需求澄清协作：指产品经理、开发团队及其他干系人如何协作开展澄清工作。

级别	主要工作完备性			人员机制	工具能力	备注
	需求澄清的时间	内容的完备性	协作			
1	在项目初期，一次性递交需求规格说明书	需求规格说明书内的内容	契约式文档传递	无	无	
2	在迭代开始之前进行需求澄清	产品经理对于用户故事的内容进行讲解，并解答团队提出的问题	召开需求澄清会	无	无	
3	同上	同上且团队确定用户故事的实现方案，识别技术风险，识别需求间的依赖和团队间的依赖。	团队内的需求澄清会，团队间的需求澄清会。	无	无	
4	同上	同上且产品经理和团队对于需求细节和验收标准达成共识，将关键信息进行记录和确认。	同上	无	企业提供的统一的协作型需求沟通工具，便于团队在澄清过程中能快速进行关键信息的更新和记录。	
5	同上	同上	同上	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人力、绩效、核算等。敏捷团队以业务价值为核心以运营为驱动的敏捷工作模式，企业为团队提供IT基础设施、基础管理等支持。	同上	

敏捷开发过程-迭代计划管理-故事与任务排期

敏捷开发将开发过程分为多个短冲刺，故事与任务的排期过程就是确定迭代冲刺目标的过程，根据产品待办列表中用户故事的优先级、依赖关系、故事规模和团队速度，确定迭代待办列表，迭代待办列表确定之后，团队成员根据优先级认领故事和任务。主要体现在三个方面：

1）排版要素：指进入排版时，信息的完备性，例如产品待办列表中用户故事的优先级、依赖关系、故事规模和团队速度等。

2）排版容量：指排版容量的大小有据可依，根据实际用户故事规模和团队速度并考虑其他影响因素后确定。

3）排版管理：指排版活动的组织形式。

级别	主要工作完备性			人员机制	工具能力
	排版要素	排版容量	排版管理		
1	产品待办清单，对产品待办清单内容的完备性不做要求	由产品经理和团队负责人根据实际需要确定，无确切依据	命令式管理，团队根据产品经理和团队负责人的要求工作。	无	无
2	产品待办清单中用户故事内容完备、优先级确定，用户故事间的依赖关系确定。	团队进行用户故事规模估算，具备团队速度的参考值	有固定的排版活动，约定为迭代开始前的固定时间，排版活动不仅确定迭代目标，同时确定迭代待办列表的优先级，便于团队在迭代开始后根据优先级顺序进行开发	无	无
3	同上	同上且具备用户故事规模估算标准	同上且具备多团队排版活动，多团队一起排版时，识别出团队间存在依赖的用户故事，约定用户故事的优先级，对于需要对齐发布周期的团队，进行对齐。	无	无
4	同上	同上	同上	无	具备工具支撑在线排版活动，能自动识别任务间的依赖，支持团队间依赖管理，能实现任务的自动流转等,对于需要进行团队对齐的情况，能自动实现团队的对齐。
5	同上	同上	同上	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限	同上

敏捷开发过程-迭代计划管理-计划变更

计划变更是指在迭代过程中，迭代目标发生变化，“响应变化胜过遵循计划”是敏捷的核心价值观之一，但进入迭代的内容发生变化会影响研发团队的工作效率，所以需要采取措施尽量减少计划变更的负面影响。主要体现在三个方面：

1）变更决策：是指决定变更和接受变更的决策方式；

2）应对变更：是指接受变更后，是否具备措施减少变更的影响；

3）减少变更：是指是否具备措施减少变更的发生。

级别	主要工作完备性			人员机制	工具能力	备注
	变更决策	应对变更	减少变更			
1	产品经理提出变更请求，变更委员会（通常为一个由需求、开发等团队负责人组成的虚拟组织）进行审批，决定是否接受变更。	无	无	无	无	
2	产品经理和团队约定计划变更的流程，产品经理提出变更请求后，与团队沟通，共同决定是否进行计划变更	发生需求变更时，团队成员决定置换的用户故事。	无	无	无	
3	同上	团队具备应对措施，减少变更带来的影响，例如：用户故事拆分时，充分考虑其独立性，减少需求变更影响的团队范围；团队在开发过程中，按照用户故事优先级进行开发；需求置换时，以小换大，即换入的用户故事规模原则上应小于换出的故事规模；优先置换出低优先级的需求；不能置换出半成品。	无	无	无	
4	同上	同上	在产品规划阶段，具备减少变更带来	无	无	

敏捷开发过程-过程管理-迭代管理

迭代管理，即贯穿于产品研发过程中以保持恒定的时长为周期，每个周期都遵从相同的框架过程，并且交付潜在的可发布最终产品增量。迭代管理主要体现在以下三个方面：

1）敏捷迭代周期：指团队能约定迭代时长、交付时长；

2）迭代协作机制：指团队内或团队间的工作进行相互配合，使得产品开发能快速交付；

3）迭代流程改进：指团队能通过不断检视迭代过程，对发现的问题能持续改进

级别	主要工作完备性			人员机制	工具能力	备注
	迭代时间周期	迭代协作机制	迭代流程改进			
1	产品分多次迭代开发，每次迭代中按照需求分析、设计、开发、上线等线性过程进行管控，完成产品部分功能	无	在下次产品完整研发过程进行改进调整	无	无	
2	团队约定任务迭代周期，约定交付周期	团队能定义清晰的活动时间、场所、参加人员；定义各类角色，明确分工，约定协作模式；约定环节间交付物、流转规则。	迭代过程问题能以用户故事形式进行改进	无	无	
3	团队内约定同上，团队间能对齐迭代计划、时间、产品集成发布时间	团队内约定同上，团队间建立协同工作机制，如通过团队间的敏捷改进会议来推进协作。	同上	无	无	
4	同上	同上	同上	无	能在团队间工作对齐、角色管理、角色工作安排、团队协作、流程数据可视化等方面提供工具支持。工具平台具备提供迭代过程的相关数据、进行分析的能力。	
5	同上	同上	同上	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建	迭代计划与企业战略相结合，建立企业级敏捷支撑平台，提供从战略规划、产品	

敏捷开发过程-过程管理-迭代活动

敏捷迭代活动，是指从产品规划、研发过程、产品交付、持续改进等维度来定义的产品迭代研发中的一系列过程，目的在于推进敏捷迭代团队的持续改进和产品的快速交付。迭代活动主要体现在以下三个方面：

1）迭代活动约定：是指团队能能在约定的时间、相对固定的场所举行相关活动；

2）迭代活动时间约定：是指团队能按照约定的时间长短进行各种会议；

3）迭代活动范围：是指团队能在各类敏捷会议中遵守约定的会议内容。

级别	主要工作完备性			人员机制	工具能力	备注
	迭代活动约定	迭代活动时间约定	迭代活动范围			
1	产品分多次迭代开发，每次迭代中按照需求分析、设计、开发、上线等线性过程进行管控，按照契约方式进行各类评审工作	时间根据会议内容确定，无约定长短	不同阶段的输出物评审	根据内容确定相关参与人员	无	
2	团队能在迭代内按照约定时间点分别完成产品计划会议、迭代计划会议、每日站立会、迭代交付评审会议、改进回顾会议。	各种会议能严格按照约定时间盒内进行	按照各类会议的要求，控制会议内容	产品经理、团队共同参与	无	
3	团队内工作方式同上，跨团队的敏捷产品开发中，多团队间建立更高级别的迭代，具有跨团队的产品代办列表。团队间定期举行跨团的计划会、评审会议、回顾会议。能不定期召开团结间协调推进会议。能对跨团队的协同问题跟进落地实施。	能对跨团队的产品按照约定时间、节奏进行验收评审会议。	同上	具备跨团队的敏捷推进协调组织，由产品经理、团队成员、跨团队的约定参与人员、及其它干系人	无	
4	在上一级的基础上，在约定周期内开展跨团队的敏捷推进会议	同上	同上	同上	无	
5	在上一级的基础上，建立企业级敏捷活动	同上	同上	企业采用扁平化的敏捷团队组织架构，赋予团队	无	

敏捷开发过程-过程管理-过程可视化及流动

通过对敏捷迭代过程的可视化展示，实时反映用户故事的迭代进展，体现产品从需求、研发、交付端到端的价值流动，通过在制品数量等工具实现价值流动的拉动式管理。过程可视化及流动主要体现在以下三个方面：

1）过程可视化：通过各种数据记录，反馈敏捷开发过程质量；

2）过程价值流动：通过各种工具体现敏捷过程的业务交付价值流动过程；

3）迭代过程改进：对数据反映的各种问题，不断改进迭代过程。

级别	主要工作完备性			人员机制	工具能力
	过程可视化	过程价值流动	迭代过程改进		
1	注重结果数据，过程数据跟踪较弱	无	无	无	无
2	团队级迭代内的过程数据进行跟踪记录，并进行可视化管理	无	无	无	
3	满足迭代数据可视化的基础上，实现端到端的可视化管理	无	无	无	通过可视化的管理工具，对产品需求收集，分析，产品故事优先级，迭代用户故事优先级等内容进行管理，实时反馈需求管理的进展
4	在满足前一级别的基础上，从产品规划到产品运营全生命周期的可视管理	通过端到端的可视化，实现产品研发的拉到式管理，能暴露过程中的问题，管理价值流动	能建立持续反馈机制，持续改进	无	通过如看板等工具进行可视化管理
5	同上	同上	同上	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人力、绩效、核算等。敏捷团队以业务价值为核心以运营为驱动的敏捷工作模式，企业为团队提供IT基础设施、基础管理等支持	在扁平化组织架构下，由企业提供过程可视化管理平台，可视化产品从用户价值提出到交付的完整过程，提供数据支撑，建立反馈，持续优化改进

敏捷开发过程-过程管理-度量分析

度量分析是对迭代过程中研发效率、质量数据进行分析，反映过程的健康程度；通过对产品端到端指标数据进行分析，实时反映产品的表现。驱动敏捷迭代的过程改进，推动企业组织架构、人员结构、财务制度等方面进行不断优化。使用敏捷迭代的方式推进改进措施的实施。度量分析主要体现在以下三个方面：

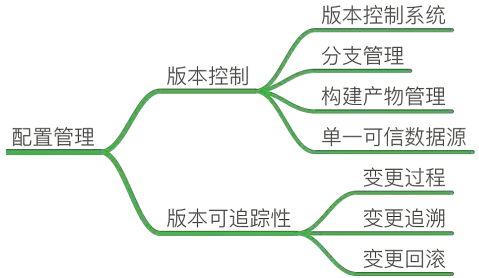
1）度量的粒度：是指敏捷迭代分析度量的详细程度；

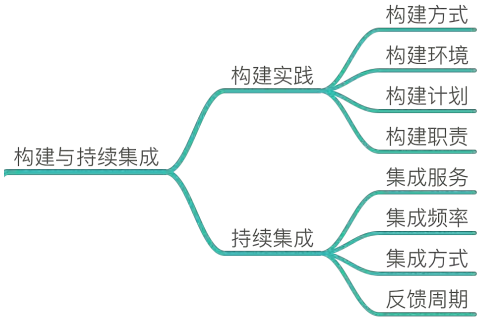
2）度量的范围：是指分析度量涉及人员组织，范围大小；

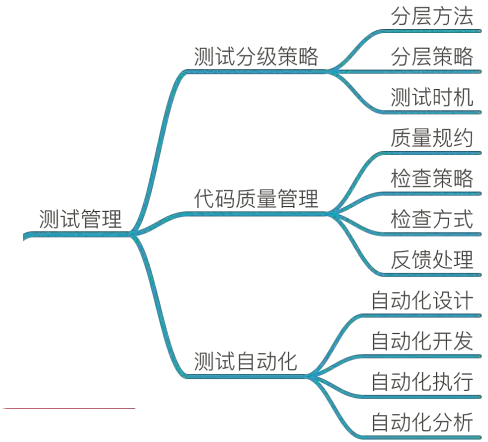
3）度量驱动持续改进：是指在分析发现问题后，能不断进行落地改进。

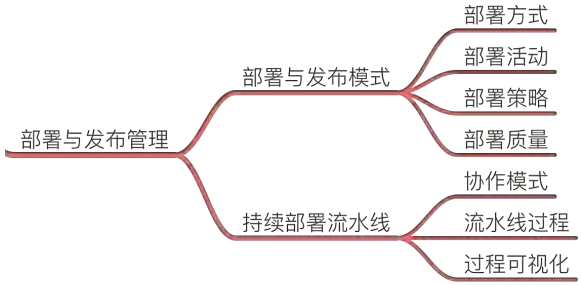
级别	主要工作完备性			人员机制	工具能力	备注
	度量粒度	度量范围	度量驱动持续改进			
1	能对团队结果数据指标进行分析跟踪。没有形成有效的过程指标分析跟踪支持	团队内	能对产品研发中质量问题进行分析，形成分析报告，对改进措施的落地推进较弱	无	无	
2	在敏捷团队中，能对迭代过程指标进行分析，从交付质量、交付速度等方面进行分析，反映研发过程的健康程度	团队内	能对团队敏捷过程发现的问题，形成团队代办任务，以敏捷迭代的方式实施改进措施	无	无	
3	在团队级的基础上，能围绕产品的完整生命周期，从需求提出、研发、交付、运营反馈等建立指标分析体系	团队内及团队间	在团队级的基础上，建立跨团队的产品级回顾会议，在会议上对多团队的迭代对齐问题，人员技能等问题进行分析，形成解决方案，以迭代的方式持续推进解决方案落地实施	无	无	
4	同上	同上	在产品级回顾会议上能以平台数据为支撑，分析敏捷过程的问题，制定改进计划，以迭代的方式持续改进	无	能通过平台支撑产品从需求到交付的端到端的过程指标展示，提供指标数据的报表分析能力，能从数据中分析发现协作的问题	
5	同上	企业级	以企业级平台数据为支撑，从战略角度分析敏捷实施过程的问题，推动企业在组织架构、人员结构、财务制度	企业采用扁平化的敏捷团队组织架构，赋予团队围绕产品自组织、自管理的权力，包括但不限于产品规划、建设、运营、人	建立企业级工具平台，能通过平台指标数据反映产品研发、公司战略、资金使用、组织效	

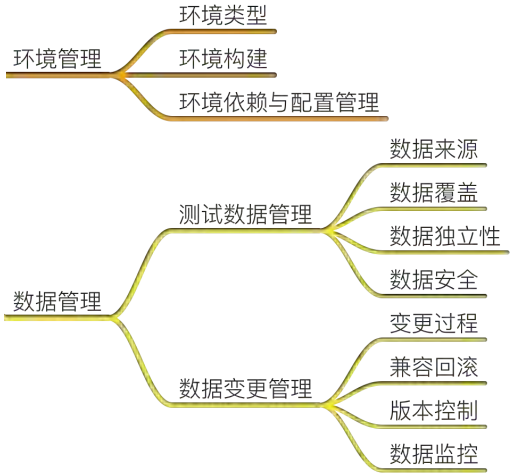
研发运营一体化能力成熟度模型 第3部分：持续交付过程

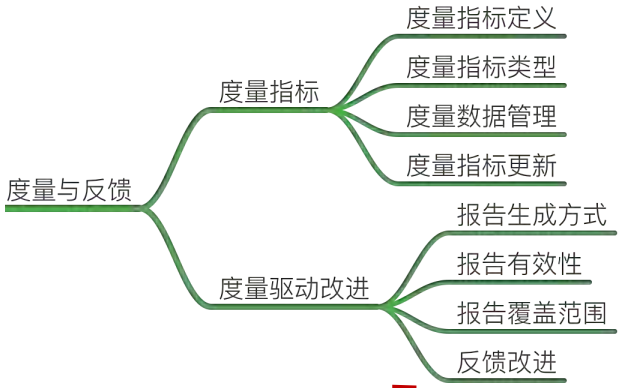












持续交付-配置管理-版本控制

级别	版本控制系统	分支管理	构建产物管理	单一可信数据源
1	未使用统一的版本控制系统，源代码分散在研发本地设备管理	缺乏明确的分支管理策略，分支生命周期混乱	未使用统一的制品库，构建产物通过直接拷贝或本地共享等方式进行分发	无
2	使用集中式的版本控制系统并将所有源代码纳入系统管理	采取长周期和大批量的方式进行代码提交，代码合并过程存在大量冲突和错误	使用统一的制品库管理构建产物，有清晰的分级和目录结构及权限管控并通过单一制品库地址进行分发	无
3	使用分布式的版本控制系统，并将所有源代码、配置文件、构建和部署等自动化脚本纳入系统管理	采取短分支频繁提交的方式，研发人员至少每天完成一次代码提交，代码合并过程顺畅	使用统一的制品库管理构建产物，并将二进制库文件和三方依赖软件工具等纳入制品库管理	版本控制系统和制品库作为单一可信数据源，覆盖生产部署环节
4	将数据库变更脚本和环境配置等纳入版本控制系统管理 版本控制系统支持自动化的变更操作	分支策略满足持续交付需求，可灵活适应产品交付	对制品库完成分级管理，有成熟的备份恢复清理策略，如采用使用分布式制品库	单一可信数据源进一步覆盖研发本地环境
5	将软件生命周期的所有配置项纳入版本控制系统管理，可完整回溯软件交付过程满足审计要求	持续优化的分支管理策略，可支持团队高效协作	同上	单一可信数据源贯穿整研发价值流交付过程 在组织内部开放共享，建立知识积累和经验复用体系

持续交付-配置管理-版本可追溯性

级别	变更过程	变更追溯	变更回滚
1	变更过程不受控且变更信息分散在每个系统内部，缺乏信息的有效共享机制	变更缺乏基本的可追溯性	变更问题定位困难且回滚操作具有高风险
2	代码变更过程应附带变更管理信息	有清晰定义的软件版本号规则，实现版本和代码的关联，可追溯版本构建对应的完整源代码信息	可支持版本间差异对比和代码级别问题定位和回滚
3	所有配置项变更由变更管理系统触发，并作为版本控制系统的强制要求	实现版本控制系统和变更管理系统的自动化关联，信息双向同步和实时可追溯	实现变更管理系统和版本控制系统的同步回滚，保证状态的一致性
4	使用同一套变更管理系统覆盖从需求到部署发布全流程	变更依赖被识别和标记，实现数据库和环境变更信息的可追溯	可根据变更管理系统按需快速导出复用软件代码变更集，如建立从变更管理系统到软件代码变更集的关系数据库
5	可视化变更生命周期，支持全程数据分析管理和满足审计要求	实现从需求到部署发布各个环节的相关全部信息的全程可追溯	支持任何时间点全部状态的自动化回滚需求

持续交付-构建与持续集成-构建实践

级别	构建方式	构建环境	构建计划	构建职责
1	采用手工方式进行构建，构建过程不可重复	使用本地设备，构建环境不可靠	没有明确的版本号规则和构建任务计划	构建工具和环境受限于团队人员能力，频繁手动干预维护
2	实现脚本自动化，通过手工配置完成构建	有独立的构建服务器，多种任务共享构建环境	明确定义版本号规则，并根据发布策略细分构建类型，实现每日自动构建	构建工具和环境由专人负责维护，并使用权限隔离
3	定义结构化构建脚本，实现模块级共享复用和统一维护	构建环境配置实现标准化，有独立的构建集群，单次构建控制在小时级	明确定义构建计划和规则，实现代码提交触发构建和定期自动执行构建	构建工具和环境由专门团队维护，并细分团队人员职责
4	实现构建服务化，可按需提供接口和用户界面用于可视化构建编排	优化构建速度，实现增量化构建和模块化构建，单次构建控制在分钟级，如可采用分布式构建集群、构建缓存等技术	分级构建计划，实现按需构建并达到资源和速度的有效平衡	构建系统服务化提供更多用户使用，构建不再局限于专业团队进行
5	持续优化的构建服务平台，持续改进服务易用性	持续改进构建性能，实现构建资源共享和动态按需分配回收，如搭建基于云服务虚拟化和容器化的分布式构建集群	同上	将构建能力赋予全部团队成员，并按需触发构建实现快速反馈

持续交付-构建与持续集成-持续集成

级别	集成服务	集成频率	集成方式	反馈周期
1	没有搭建持续集成服务，团队成员缺乏对持续集成的理解	长期本地开发代码集成频率几周或者几月一次	代码集成作为软件交付流程中的一个独立阶段	每次集成伴随大量的问题和冲突，集成期间主干分支长期不可用
2	搭建统一的持续集成服务并对系统进行日常维护和管理	采用团队定期统一集成的策略，代码集成频率几天或者几周一次	在部分分支上进行每天多次的定时构建	集成问题反馈和解决需要半天或者更长时间
3	组建专门的持续集成团队，负责优化持续集成系统和服务	研发人员至少每天向代码主干集成一次	每次代码提交触发自动化构建，构建问题通过自动分析精准推送相关人员处理	集成问题反馈和解决可以在几个小时内完成
4	持续集成嵌入每个研发团队日常活动，实现持续集成系统服务化和自助化	研发人员每天多次向代码主干集成，每次集成代价较低	每次代码提交构建触发自动化测试和静态代码检查，测试问题自动上报变更管理系统，测试结果作为版本质量标准要求，如：采取质量门禁等方式强化主干代码质量	集成问题反馈和解决控制在30分钟以内完成
5	持续优化和改进团队持续集成服务，实现组织交付能力提升	任何变更（代码，配置，环境）都会触发完整的持续集成流程	实现持续集成分级和自动化测试分级，满足不同模块和集成阶段的差异化需求	集成问题反馈和解决控制在10分钟以内完成

持续交付-测试管理-测试分层策略

GOPS2017 Shanghai

级别	分层方法	分层策略	测试时机
1	只进行用户/业务级的UI测试	尚未建立测试分层策略，测试不分层	测试在软件交付过程中在开发完成后才介入
2	采用接口/服务级测试对模块/服务进行覆盖全面的接口测试；采用代码级测试对核心模块的函数或类方法进行单元测试；对系统进行基本的性能测试	测试开始分层，但对测试分层策略缺乏系统的规划，对用户/业务级测试、接口/服务级、代码级测试分布比例由高到低，各层测试缺乏有效的设计	测试在持续交付过程中的介入时间提前到开发的集成阶段，接口/服务级测试在模块的接口开发完 成后进行
3	采用代码级测试对模块的函数或类方法进行覆盖全面的单元测试；系统全面的进行性能、容量、稳定性、可靠性、易用性、兼容性、安全性等非功能性测试	对测试分层策略进行系统的规划，用户/业务级、接口/服务级、代码级测试分布比例由低到高，充分设计；对非功能性测试进行全面系统的设计	测试在持续交付过程中的介入时 间提前到开发的编码阶段，代码级测试在模块的函数或类方法开发完成后进行
4	采用测试驱动开发的方式进行代码级、接口级测试；采用探索性测试方法对需求进行深入挖掘测试	测试分层策略的各层测试具有交叉互补性	代码级测试在模块的函数或类方法开发过程中同步进行和完成；接口/服务级测试在模块的接口开发过程中同步进行和完 成
5	采用验收测试驱动开发的方式进行用户/业务级的UI 测试	定期验证测试分层策略，是否完整、有效，持续优化策略	在需求阶段进行用户/业务级测试设计，在需求特性开发、交付 整个过程中同步进行并完成测试

持续交付-测试管理-代码质量管理

级别	质量规约	检查策略	检查方式	反馈处理
1	代码质量检查无任何规约	代码质量检查无针对检查范围、质量门限等相关的策略	代码质量检查采用人工方式进行评审	对代码质量检查结果处理不及时，遗留大量技术债
2	代码质量检查具备基本规约，但还缺乏完整性和有效性	代码质量检查有针对检查范围、质量门限的策略，对代码规范、错误和圈复杂度、重复度等质量指标进行检查分析	代码质量检查采用自动化结合手工方式进行	对代码质量检查结果给出反馈，根据反馈进行处理，对遗留的部分技术债缺乏跟踪和管理，导致遗漏
3	代码质量检查具备完整、有效和强制执行的规约	代码质量检查将安全漏洞检查、合规检查纳入到检查范围	代码质量检查完全自动化，不需要手工干预	根据代码质量检查结果反馈及时处理，技术债仍有短期遗留，但进行有效的跟踪、管理和处理
4	代码质量检查规约根据需要可进行扩展和定制	代码质量检查针对检查范围、质量门限的策略可根据需要灵活调整	对代码质量检查发现的部分问题自动提出修改建议，支持可视化	将检查结果强制作为版本质量标准要求，根据代码质量检查提出的修改建议，对问题及时处理，在研发阶段主动解决技术债
5	定期验证代码质量规约的完整性和有效性，持续优化	定期验证代码质量策略的完整性和有效性，持续优化	具备企业级的代码质量管理平台，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，可有效追溯并对代码质量进行有效度量

持续交付-测试管理-自动化测试

级别	自动化设计	自动化开发	自动化执行	自动化分析
1	未采用自动化方式测试，纯手工测试	尚未对自动化测试脚本进行开发和管理，手工测试	手工测试执行效率低下，以周级为单位	手工对测试结果进行分析判断，错误高，可信度低
2	尚未对测试用例中自动化部分进行规划和设计，覆盖不完整	对自动化测试脚本进行开发和本地管理	对用户/业务级测试采用 自动化测试，自动化测试的执行效率不高，以天级为单位	对自动化测试结果具备一定的自动判断能力，存在一定的误报，可信度不足
3	根据需求、接口和代码对不同测试分层中自动化测试用例进行规划和设计，自动化覆盖比较完整	自动化测试脚本开发采用数据驱动、关键字驱动等方法；使用版本控制系统对自动化测试脚本进行有效管理	从代码级、接口级到UI级测试实现了端到端的自动化测试打通；自动化测试执行效率较高，代码级测试分钟级，UI级测试小时级	对自动化测试结果具备较强的自动判断能力，误报少，可信度高
4	对性能、稳定性、可靠性、安全性等非功能性测试中自动化用例进行规划和设计，自动化覆盖完整	自动化测试用例脚本间具备独立性和大批量执行的健壮性	有组织级的统一自动化测试平台，和上下游需求、故障系统打通；可以根据需求针对性自动关联选择自动化测试用例脚本执行；可以将由于版本原因导致的失败用例和故障关联	自动化测试数据模型标准化，和上下游需求、故障等研发数据关联，可以对自动化测试效果进行度量分析。例如：需求测试覆盖率、测试通过率和测试效率等。
5	对故障和测试进行复盘，对遗漏的测试用例进行补充，不断优化和完善，持续提升覆盖率	自动化脚本是测试用例设计的活文档，自动化脚本开发和测试用例设计完全统一	采用企业级统一的自动化测试平台，以云化的方式提供测试服务，进行分布式测试调度执行，提高测试执行效率和资源利用率；定期验证自动化执行策略，持续优化	对自动化测试结果可以智能分析，自动分析失败用例的失败类型及原因，可以自动向故障管理系统提交故障，可信度高

持续交付-部署与发布管理-部署与发布模式

GOPS2017 Shanghai

级别	部署方式	部署活动	部署策略	部署质量
1	运维人员手工完成所有环境的部署	部署过程复杂不可控，伴随大量问题和较长的停机时间	采用定期大批量部署策略	部署整体失败率较高，并且无法实现回滚，生产问题只能在线上修复，修复时间不可控
2	运维人员通过自动化脚本实现部署过程部分自动化	部署过程通过流程文档定义实现标准化整体可控	应用作为部署的最小单位，应用和数据库部署实现分离，实现测试环境的自动化部署	实现应用部署的回滚操作，部署失败率中等，问题可及时修复
3	部署和发布实现全自动化，同时支持数据库自动化部署	使用相同的过程和工具完成所有环境部署，一次部署过程中使用相同的构建产物	可运行的环境作为部署的最小单位，应用和配置进行分离	部署活动集成自动化测试功能，并以测试结果为部署前置条件 每次部署活动提供变更对象范围报告和测试报告
4	部署发布服务化，实现交付团队自助一键式多环境自动化	部署过程可灵活响应业务需求变化，通过合理组合高效编排	通过多种部署发布策略保证流程风险可控，如：蓝绿部署，金丝雀发布	建立监控体系跟踪和分析部署过程，出现问题自动化降级回滚，失败率较低
5	持续优化的部署发布模式和工具系统平台	持续部署，每次变更都触发一次自动化生产环境部署过程	软件交付团队自主进行安全可靠的部署和发布活动	持续优化的部署监控体系和测试体系，部署失败率维持在极低水平

持续交付-部署与发布管理-持续部署流水线

级别	协作模式	流水线过程	过程可视化
1	整个软件交付过程严格遵循预先计划，存在复杂的部门间协作和等待，只有在开发完成后才进行测试和部署	软件交付过程中的大部分工作通过手工方式完成	交付过程中的信息是封闭的，交付状态不可追溯
2	通过定义完整的软件交付过程和清晰的交付规范，保证团队之间交付的有序	软件交付过程中的各个环节建立自动化能力以提升处理效率	交付过程在团队内部可见，信息在团队间共享，交付状态可追溯
3	团队间交付按照约定由系统间调用完成，仅在必要环节进行手工确认	打通软件交付过程中的各个环节，建立全流程的自动化能力，并根据自动化测试结果控制软件交付质量	交付过程组织内部可见，团队共享度量指标
4	团队间依赖解耦，可实现独立安全的自主部署交付	建立可视化部署流水线，覆盖整个软件交付过程，每次变更都会触发完整的自动化部署流水线	部署流水线全员可见，对过程信息进行有效聚合分析展示趋势
5	持续优化的交付业务组织灵活响应业务变化改善发布效率	持续部署流水线驱动持续改进	部署流水线过程信息进行数据价值挖掘，推动业务改进

持续交付-环境管理

级别	环境类型	环境构建	环境依赖与配置管理
1	环境类型只有生产环境和非生产环境的划分	环境的构建通过人工创建完成	无依赖管理，环境的管理就是一个OS的交付
2	IT交付过程意识到部分测试环境的重要性，开始提供功能测试环境。	环境构建通过一键化的脚本或者虚拟机来完成的，构建过程完全黑盒化完成。	以应用为中心有OS级别的依赖和配置管理能力，比如说操作系统版本、组件版本、程序包版本等等
3	持续交付过程意识到研发环境的重要性，开始提供面向各类开发者独立的研发工作区。	环境的构建通过资源交付平台来完成，并且底层是由云来交付	以应用为中心，有服务级依赖的配置管理能力，比如说依赖的关联服务，Mysql服务、cache服务、关联应用服务等等
4	全面的测试与灰度环境对于质量交付过程来说非常重要，有各类的环境类型划分，区分了开发者，技术测试及业务测试环境以及灰度发布环境等等	环境的构建可以通过Docker容器化快速交付，低成本构建一个新的环境	环境和依赖配置管理可以资源化描述，类似dockerfile，大大提升其配置管理能力
5	根据业务与应用的需要，弹性分配各类环境	环境的构建结合底层IT资源状况，采用了各类混合IT技术，根据业务及应用架构弹性构建	环境依赖和配置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的变化而变化

持续交付-数据管理-测试数据管理

级别	数据来源	数据覆盖	数据独立性	数据安全
1	每次测试时手工创建数据，测试数据都是临时性的	测试数据覆盖率低，仅支持部分测试场景，无法有效支持测试工作	测试数据没有版本控制和备份恢复机制	测试数据来源复杂，混入核心生产数据，带来信息安全风险
2	从生产环境导出一个子集并进行清洗后，形成基准的测试数据集，满足部分测试用例执行要求	测试数据覆盖主要场景，包括正常类型，错误类型以及边界类型，并进行初步的分类分级，满足不同测试类型需要	测试数据有明确备份恢复机制，实现测试数据复用和保证测试一致性	测试数据经过清洗，不包含敏感信息，有效避免信息安全风险
3	同上	建立体系化测试数据，进行数据依赖管理，覆盖更加复杂的业务场景	每个测试用例拥有专属的测试数据，有明确的测试初始状态 测试用例的执行不依赖其他测试用例执行所产生的数据	同上
4	每个测试用例专属的测试数据都可以通过模拟或调用应用程序API的方式自动生成	测试数据覆盖安全漏洞和开源合规等需求场景并建立定期更新机制	通过测试数据分级，实现专属测试数据和通用测试数据的有效管理和灵活组合，保证测试数据的独立性	同上
5	所有的功能、非功能测试的测试数据，都可以通过模拟、数据库转储或调用应用程序API的方式自动生成	持续优化的持续数据管理方式和策略	同上	同上

持续交付-数据管理-数据变更管理

级别	变更过程	兼容回滚	版本控制	数据监控
1	数据变更由专业人员在后台手工完成 数据变更作为软件发布的一个独立环节，单独实施和交付	没有识别数据库和应用版本，存在不兼容风险	数据变更没有纳入版本控制，变更过程不可重复	没有建立变更监控体系，变更结果不可见
2	数据变更通过文档实现标准化，使用自动化脚本完成变更	建立数据库和应用的版本对应关系，并跟踪变更有效性	数据变更脚本纳入版本控制，并与数据库版本进行关联	对变更日志进行收集分析，帮助问题快速定位
3	数据变更作为持续部署流水线的一个环节，随应用的部署自动化完成，无需专业人员单独执行	每次数据变更同时提供明确的恢复回滚机制，并进行变更测试，如：提供升级和回滚两个自动化脚本	同上	对数据变更进行流程分级定义，应对不同环境下的高危操作
4	应用程序部署和数据库迁移解耦，可单独执行	数据变更具备向下兼容性，支持保留数据的回滚操作和零停机部署	同上	对数据变更进行监控，自动发现异常变更状态
5	持续优化的数据管理方法，持续改进数据管理效率	同上	同上	监控数据库性能并持续优化

持续交付-度量与反馈-度量质量

级别	度量指标定义	度量指标类型	度量数据管理	度量指标更新
1	度量指标没有明确定义，对度量价值的理解是模糊的	无	度量数据是临时性的，没有收集管理	无
2	在持续交付各个阶段定义度量指标，度量指标局限于职能部门内部	度量指标以结果指标为主，如变更频率，需求交付前置时间，变更失败率和平均修复时间，	度量数据的收集是离散的不连续的，历史度量数据没有进行有效管理	度量指标的设立和更新是固化的，度量指标没有明确的优先级
3	建立跨组织度量指标，进行跨领域综合维度的度量	度量指标覆盖过程指标，客观反映组织研发现状	度量数据的收集是连续的，历史度量数据有明确的管理规则	度量指标的设立和更新是动态的，可以按照组织需求定期变更，度量指标的优先级在团队内部可以达成一致
4	整个研发团队共享业务价值导向的度量指标，实现指标的抽象分级，关注核心业务指标	度量指标覆盖探索性指标，关注展示趋势和识别潜在改进	度量数据的收集是连续且优化的，对历史数据数据进行有效的挖掘分析	建立完整的度量体系和成熟的度量框架，度量指标的设立和更新可按需实现快速定义并纳入度量体系，推动流程的持续改进
5	持续优化的度量指标，团队自我驱动持续改进	支持改进目标和试验结果的有效反馈，用于经验积累和指导下一阶段的改进工作	同上	度量指标可基于大数据分析和人工智能自动识别推荐，并且动态调整指标优先级

持续交付-度量与反馈-度量驱动改进

级别	报告生成方式	报告有效性	报告覆盖范围	反馈改进
1	度量报告通过手工方式生成，没有标准化的格式定义，内容缺乏细节	数据时效性无法保证	受众局限于报告生成人员及相关的小范围内部	报告发现的问题没有进行有效跟踪落实，问题长期无法改进
2	度量报告以自动化方式生成，通过预定义格式和内容标准化度量报告	数据体现报告生成时间点的最新状态	由预先定义的事件触发自动化报告发送，受众覆盖团队内部成员	测试报告中反馈的问题录入问题追踪系统，进行持续跟踪
3	度量报告进行分类分级，建立多种度量反馈渠道，内容按需生成	通过可视化看板实时展示数据	实现报告精准范围推送，支持主动订阅，受众覆盖跨部门团队	度量反馈问题纳入研发迭代的待办事项，作为持续改进的一部分
4	建立跨组织级统一的数据度量平台，数据看板内容可定制	通过可视化看板聚合报告内容，自动生成趋势图，进行趋势分析	多维度产品状态实时信息展示	度量反馈的持续改进纳入研发日常工作，预留时间处理非功能性需求和技术债务，并且识别有效改进并扩展到整个组织，作为企业级知识体系积累保留
5	持续优化的度量方法，平台和展现形式	同上	同上	通过数据挖掘实现跨组织跨流程数据度量分析，分析结果作为业务决策的重要依据，帮助组织持续改进价值交付流程

DevOps 标准：样例

目 次
前 言
1. 范围
2. 规范性引用文件
3. 术语
4. 缩略语
5. 总体架构
6. 分级技术要求
6.1 敏捷开发管理
6.2 持续交付
6.2.1 配置管理
6.2.1.1 版本控制
6.2.1.2 版本可追溯性
6.2.2 构建与持续集成
6.2.3 测试管理
6.2.3.1 测试分层策略
6.2.3.2 代码质量管理
6.2.3.3 自动化测试
6.2.4 部署与发布管理
6.2.5 环境管理
6.2.6 数据管理
6.2.7 度量与反馈
6.3 应用运维
6.4 应用架构
6.5 组织与文化
6.2.4 部署与发布管理
6.2.4.1 部署与发布模式
级别
部署方式
部署活动
部署策略
部署质量
6.2.4.2 持续部署流水线

DevOps 标准：工作机制

标准输出单位：中国通信标准化协会 TC1 WG5 云计算工作组
标准编写工作组：云计算开源产业联盟 高效运维社区

单位	姓名	职务
中国信息通信研究院	栗蔚	主任工程师
	牛晓玲	工程师
高效运维社区、DevOps时代社区	萧田国	联合发起人
	张乐	联合发起人
	景韵	联合发起人
腾讯	刘栖铜	总经理助理
	党受辉	技术总监
	梁定安	技术副总监
阿里游戏	李运华	技术专家
京东	徐奇琛	技术总监
YY直播	韩方	技术专家
新乐视	石雪峰	技术总监
中兴通信	闫林	中兴学院副院长
中兴通信	鞠炜刚	测试总监

单位	姓名	职务
上海仪电	杨天顺	中央研究院常务副院长
	刘超	中央研究院技术总监
中国移动浙江公司	方炜	总监
	李海传	架构师
	廖希密	工程师
中国移动南方基地	林恩华	网管支撑中心 云计算运维负责人
	燕杰	网管支撑中心 运维专家
中国电信集团公司	陈靖翔	运营维护处副处长
中国银行	张欣	质量经理
中国银联	任明	运维架构主管
中国太平洋保险集团	胡罡	应用运行支持部副总经理
华泰证券	樊建	技术总监
优维科技	王津银	创始人

培训

高效运维社区
GreatOPS Community

评估

云计算开源产业联盟
OpenSource Cloud Alliance for industRy,OSCAR

DevOps 标准：下载链接

云计算开源产业联盟

OpenSource Cloud Alliance for IndustRy,OSCAR

GOPS2017
Shanghai

进一步推动国内云计算开源产业健康快速发展

Thanks

云计算开源产业联盟

高效运维社区

荣誉出品