

**Technical Team Building Solutiongs**

**技术团队建设方案**

**技术团队建设方案**

**张俊武**

目录

[一、 方案背景 4](#_Toc6526)

[二、 团队管理 5](#_Toc22218)

[2.1团队分工 5](#_Toc11152)

[2.2敏捷团队 6](#_Toc18097)

[2.2.1敏捷or瀑布模式 6](#_Toc10912)

[2.2.2 起个响亮的名称代号 9](#_Toc6604)

[2.3团队OKR 10](#_Toc2699)

[2.4团队公约 12](#_Toc4088)

[2.5新人融入 14](#_Toc354)

[2.6上线流程 14](#_Toc19970)

[2.7团队分享 16](#_Toc21434)

[2.8团队周报 18](#_Toc21821)

[三、 研发团队 19](#_Toc22624)

[3.1开发流程规范 19](#_Toc14563)

[3.2 开发代码规范 20](#_Toc5826)

[3.3 开发分支规范 21](#_Toc1886)

[四、 产品团队 22](#_Toc15758)

[4.1 产品职责分工 22](#_Toc20198)

[4.2 知识库 23](#_Toc23378)

[4.3 业务全景图 24](#_Toc8225)

[五、 测试团队 25](#_Toc20777)

[5.1 测试流程规范 25](#_Toc18082)

[5.2 线上质量看板 26](#_Toc30644)

[六、 运维团队 27](#_Toc28281)

[6.1 平台手册维护 27](#_Toc28910)

[6.2 工单维护 27](#_Toc26420)

[6.3线上事故复盘 28](#_Toc13730)

[七、 架构团队 30](#_Toc30725)

[7.1 成员组成 30](#_Toc9097)

[7.2 规范制定 30](#_Toc5602)

[7.3 团队职责 31](#_Toc14682)

[7.4 架构设计流程规范 32](#_Toc5082)

[阶段1：架构愿景 34](#_Toc22608)

[阶段2：业务架构 34](#_Toc26366)

[阶段 3：信息系统架构 34](#_Toc24309)

[阶段 4：技术架构 34](#_Toc8834)

[阶段 5：机会和解决方案 34](#_Toc20273)

[阶段 6：迁移规划 34](#_Toc13836)

[阶段 7：实施治理 34](#_Toc500)

[阶段 8：架构变更管理 35](#_Toc1860)

[核心：需求管理检查 35](#_Toc15593)

[7.5代码评审规范 35](#_Toc2343)

[八、 总结 36](#_Toc25372)

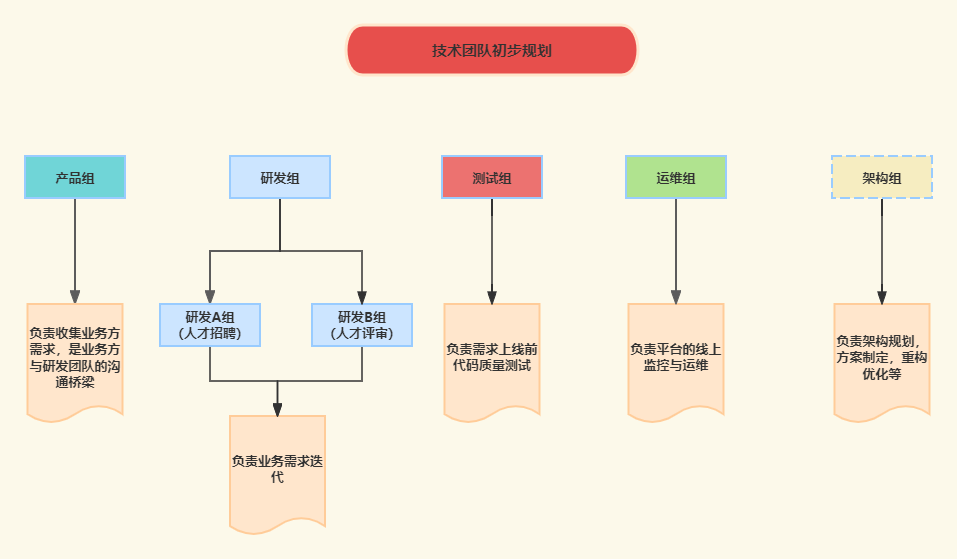
# 方案背景

国内很多平台搭建之初并没有进行很好的架构规划与相应团队管理建设，随着业务的不断拓展与壮大，原平台的技术与需求迭代速度逐渐跟不上业务发展的脚步，技术平台逐渐成为业务的掣肘。另一方面来说，缺少架构全景与重构优化，原有的架构会逐渐变的臃肿，存在千里之堤溃于蚁穴的长远隐患。

以下是本人查阅公司相关资料，结合个人以往的架构与管理经验，提交的一份初步的技术团队建设方案。

# 团队管理

## 2.1团队分工

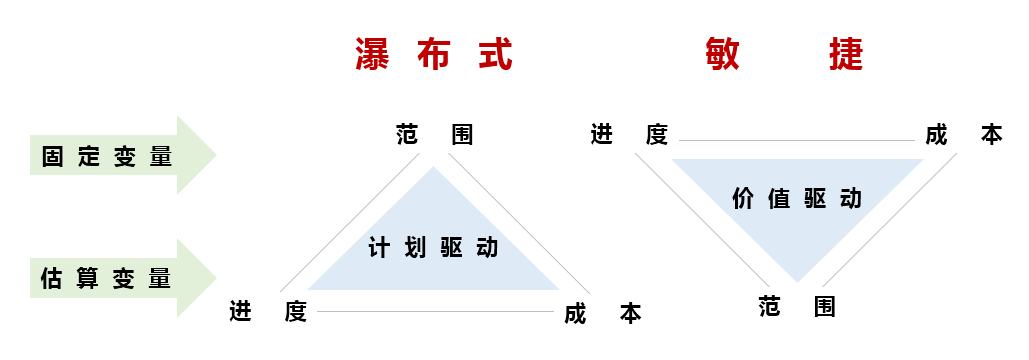


初步的技术团队规划分工如下：

* **产品组**：负责收集业务方需求，是业务方与研发团队的沟通桥梁。
* **研发组**：负责业务需求迭代。
* **测试组**：负责对研发同学提测的代码质量进行把控。
* **运维组**：负责平台的线上监控与运维。
* **架构组（虚线）**：从研发小组中挑选技术能力强的同学组成，负责架构规划、代码评审、规范制定等。

## 2.2敏捷团队

### 2.2.1敏捷or瀑布模式



#### 1.开发流程不同

瀑布开发采用线性的开发流程，按照预先规划的顺序依次进行需求分析、设计、编码、测试和维护等环节。每个环节都有明确的交付物和里程碑。开发团队在完成上一个环节后才能进入下一个环节。

敏捷开发采用迭代和增量的开发方式。开发工作被划分为短期的迭代周期，每个迭代周期通常持续数周到数月。在每个迭代周期中，团队会完成一部分功能的开发，并进行测试和评审。随着迭代的进行，产品逐渐完善。

#### 2.需求变更容忍度不同

瀑布开发对需求变更的容忍度较低。在开发过程中，需求的变更会导致整个开发计划的重新规划和调整。因此，在瀑布开发中，需求往往需要在项目开始之前充分明确和稳定。

敏捷开发更加灵活，对需求变更有更高的容忍度。敏捷开发团队能够在每个迭代周期中灵活地接受和适应需求的变化，通过不断的反馈和迭代，及时调整开发方向，以满足客户的需求。

#### 3.交付时间不同

由于瀑布开发的线性流程，整个项目需要在一开始就制定详细的计划和时间表。项目交付通常在开发周期的最后阶段才完成，因此交付时间较长。

敏捷开发的迭代周期较短，团队能够在每个迭代周期中交付可用的软件功能。客户可以在开发过程中不断审视和验证软件，并提供及时的反馈，这有助于缩短交付时间。

#### 4.沟通与合作方式不同

瀑布开发通常采用较为正式的沟通方式，如书面文档和正式会议。沟通和合作主要在开发阶段之间进行，较少涉及客户和用户。而敏捷开发更加注重团队内外的沟通和合作。团队成员之间会频繁交流，以确保团队共同理解需求，并解决可能出现的问题。客户和用户也会被积极地纳入到开发过程中，参与需求讨论和功能验证。

#### 5.风险管理方式不同

瀑布开发在项目开始之前会进行全面的风险评估，并尽可能规避和减轻风险。由于开发计划一经确定，变更较为困难，因此瀑布开发的风险管理主要集中在前期。

相比之下，敏捷开发更加注重持续的风险管理。团队在每个迭代周期中都会进行风险评估，并根据评估结果调整开发策略和计划。通过迭代的方式，敏捷开发能够更及时地应对风险，并减少项目失败的风险。

#### 6.优选敏捷开发模式

综合上文所述，瀑布开发和敏捷开发在开发流程、需求变更、交付时间、沟通与合作以及风险管理等方面存在明显的差异。考虑到自己公司项目的特点和需求，需要不断的提升平台能力与迭代需求，选择敏捷开发模式比较合适。

### 2.2.2 起个响亮的名称代号



不论是蓝图规划、重大需求开发建设，还是大型专项升级改造，起个名称代号（别名），会给团队成员提供附加的信息暗示，除此之外，名称代号有助于提升任务的重要性和战略高度，获得上级领导更大的重视和资源倾斜。对于拉动任务的执行，其意义不可小视。

如果平台的重要性足够高，参与其建设的人数足够多，起个名称代号会是个好的选项，对于平台本身来说增加了一层语义信息，也是一种赋能，可以从影响力视角思考这个问题。

我们业务方向有两个，匹配各自相应的研发团队，就可以为每个研发团队起一个响亮的名称代号，可以由组内同学自己投票选举，例如聚火、筋斗云等等。

## 2.3团队OKR



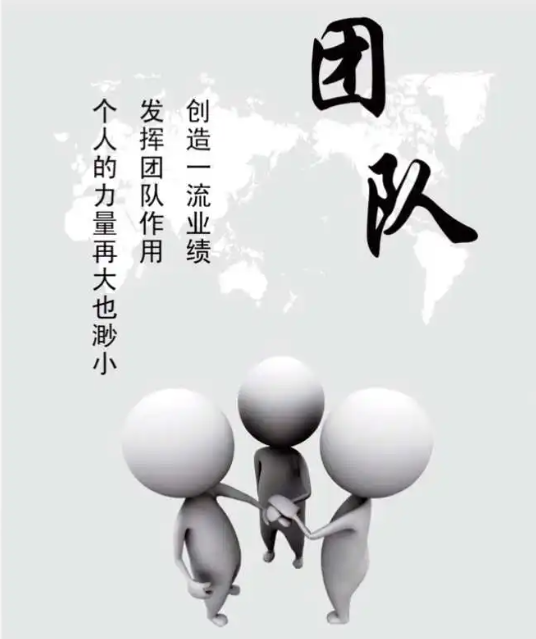
在每个季度之初，技术团队成员需要制定自己下一个季度的OKR，OKR（Objectives and Key Results）即目标与关键成果法，是一套明确和跟踪目标及其完成情况的[管理工具](https://baike.baidu.com/item/%E7%AE%A1%E7%90%86%E5%B7%A5%E5%85%B7/9143974?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/OKR/_blank)和方法。OKR的主要目标是明确公司和团队的“目标”以及明确每个目标达成的可衡量的“关键结果”。

团队成员需要明确自己在下一个季度中预定可以达成的目标，比如完成多少个核心需求、出现的bug数量少于多少等。

或者可以根据团队OKR拆解自己OKR，并按照OKR的方向努力。



## 2.4团队公约



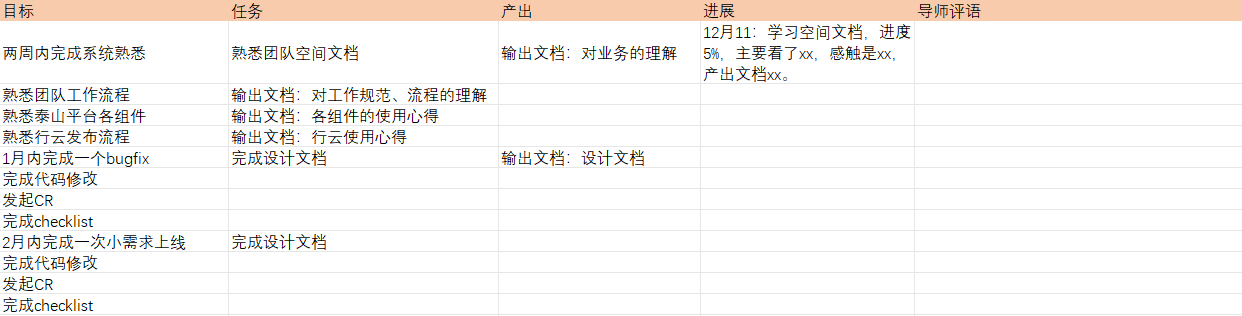
初步公约如下，后续保持迭代：

* **OKR**：根据团队OKR拆解自己OKR，并按照OKR的方向努力。
* **绩效**：获得绩效A，除了完成本职工作外，还需要对团队OKR有明显贡献，且贡献自己能够量化的说清楚。
* **规范**：严格遵守规范清单，不合适主动沟通，不沟通就遵守。
* **敏捷**：真正理解敏捷团队的精髓，不断提升敏捷成熟度。
* **工程质量**：对线上有高度敬畏感，尽可能少的线上bug，尽量避免线上故障。
* **规章制度**： 按公司要求完成请假须提前一天完成流程，并在相关群进行周知。
* **结果导向**：没结果就别说过程了；有结果，过程要讲清楚；不糊弄结果，一般站不住脚。
* **职能定义**：产品聚焦产品规划和功能实现逻辑，研发聚焦功能实现方案和质量，测试聚焦工程质量把控，运维聚焦快速解决用户问题。
* **成长**：公司有增长目标和淘汰机制，每个个体只要成长，才能支撑团队跟得上公司节奏，要提升写作基本功、代码基本功，要提升自己的知识体系，多看书是一个好的方法。
* **信任**：默认有信任，一些具体事情做不好信任变弱，比如承诺了做不到、线上老出问题。你做了让组织不信任的事情，就要承担组织对你的不认可和结果。

## 2.5新人融入



新入职人员需要制定自己融入团队的短期计划表，比如多长时间内熟悉某一块业务代码，多长时间内可以独立完成某一个需求迭代。



## 2.6上线流程

**研发铁律，必须遵守**：

1.上线时间参照《上线管理规范》，其它时间的上线定义为紧急上线，走紧急上线流程，尽量遵守上线时间。

2.上线前代码必须经过自测及CodeReview，需求及项目必须提测。

3.上线前必须先经过预发验证，测试人员需参与预发验证，涉及上下游的改动必须经过联调。

4.上线前必须检查“上线前检查表”中的各项内容，线上配置文件的修改必须双人交叉检查，抽取出来的配置文件修改时应先修改代码文件中的相关配置，再修改抽取出来的配置。

5.上线前必须准备可执行的上线回退方案。

6.上线前评估是否支持灰度，如果支持必须灰度发布，全量发布可延至次日20：00以后，不定义为紧急上线。

7.上线前必须通知相关人员，上线结果及时反馈，上线成功以产研测三方确认为准。

8.上线中必须先发布一台实例，观察实例启动是否正常，日志中是否有异常，服务是否上线，服务可用率以及JVM各项参数是否正常，一切正常后串行或并行30%发布。

9.上线后必须至少监测半个小时，监测参考“线上发布后验证项”，较大规模的上线需安排在次日早再次监测。

10.上线后发现问题遵守“先回退，再通知，再分析，再解决”原则。

11. 新依赖接口上线前要检查线上分组配置（避免预发分组配置到线上）、超时时间和重试次数，上线后要及时添加报警设置（服务端）。

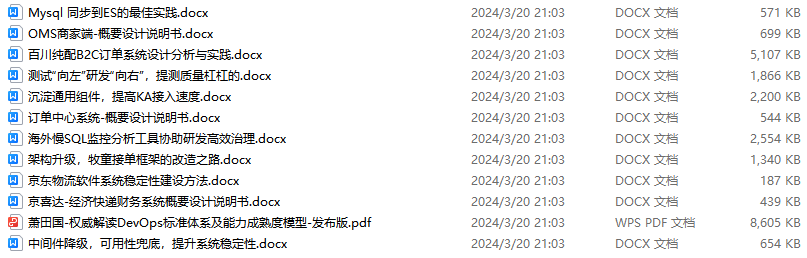


## 2.7团队分享



平台与团队在不断成长的过程中，需要注意相应技术资产的存档，建立一套完备的资产管理方法，这样可以避免诸多项目风险，还可以提升技术团队的能力与建设。

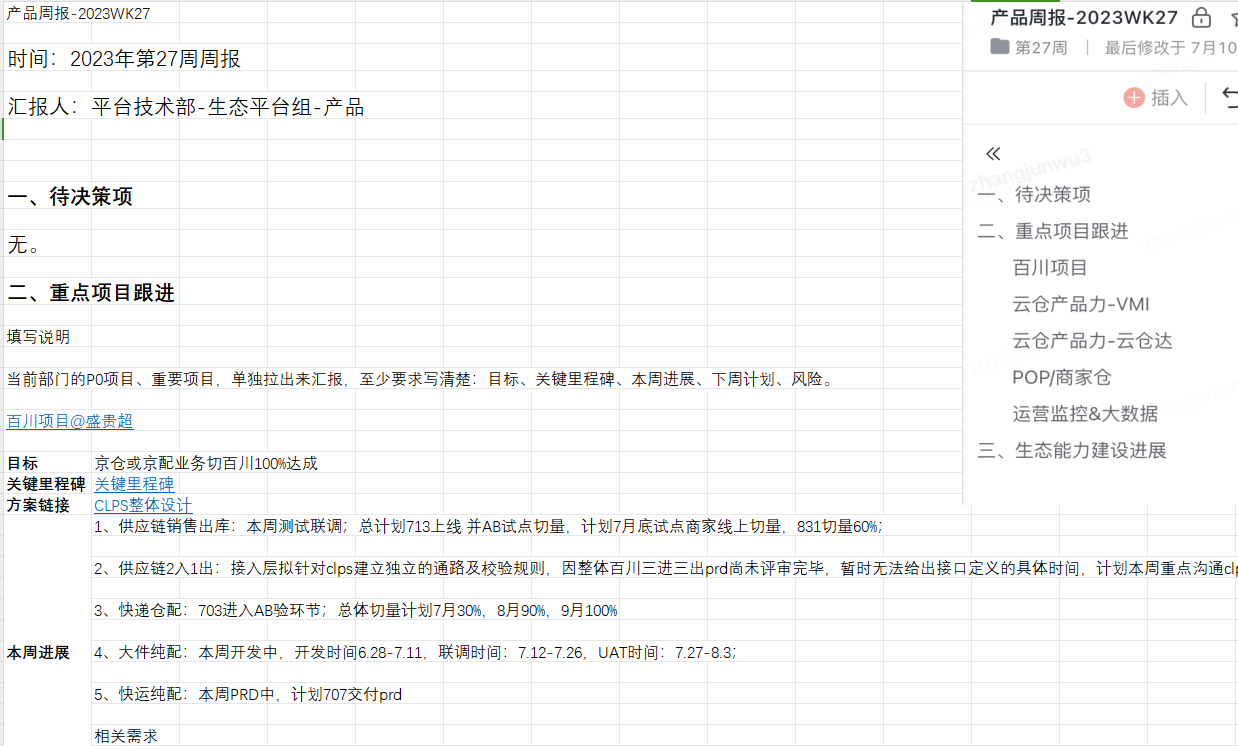
团队成员可以将以往的业务经验或者自己对哪一项技术掌握比较深刻，都可以主动申请在团队内部进行分享会议，促成团队整体能力提升。



## 2.8团队周报

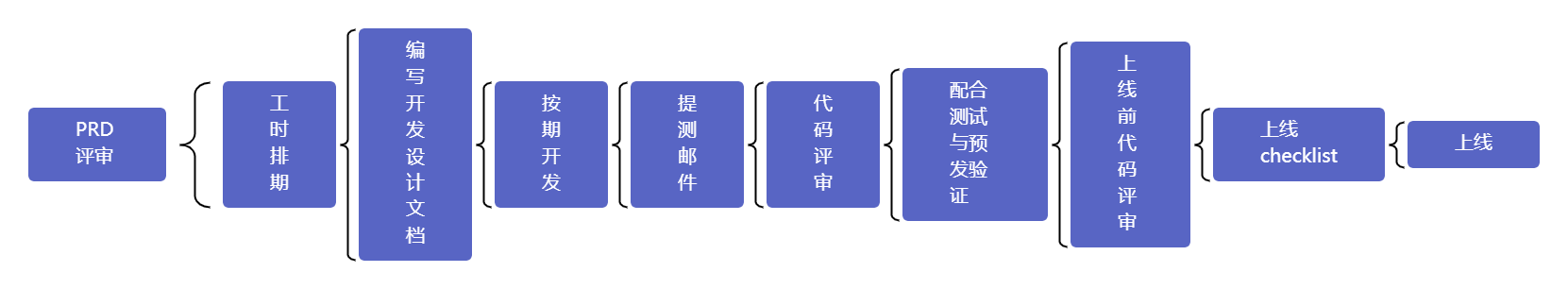


周报的意义在于团队成员对自己一周内的工作进行量化展出，并预期自己下周的工作规划，可显著提升团队效能。



# 研发团队

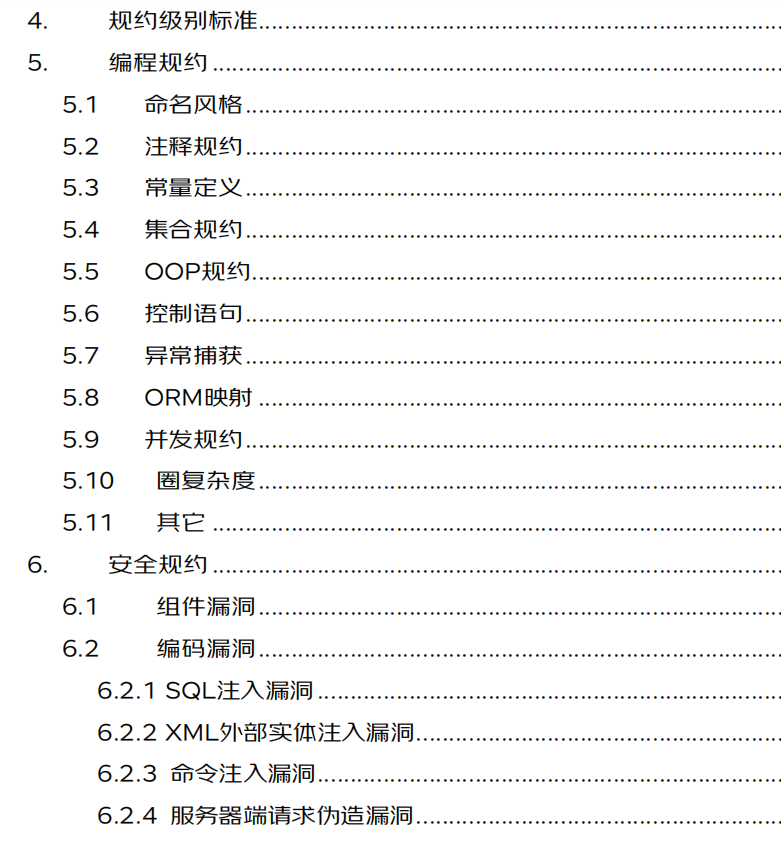
## 3.1开发流程规范



* **PRD评审**：研发需要参与PRD评审，且在评审会中对于存疑的点与产品进行沟通，做到不留死角，不懂就问，需要与产品达成目标一致性。
* **工时排期**：研发理解PRD之后，需要根据自己的能力与需要开发的需求工程量进行评估，得出需要的开发工时。
* **编写开发设计文档**：研发在正式开始写代码前，需要对需要的改动与自己的设计进行阐述，并由架构小组成员进行设计文档评审。
* **按期开发**：研发需要在预计工时内完成相应的代码开发，如若延期，需要提前报项目风险，并对延期原因进行阐述，重新制定工时。
* **提测**：研发开发完成后，需要先进行自测，自测通过后，发起提测邮件通知测试组。
* **代码评审**：研发提测后，需要通知架构小组进行第一次代码评审，架构小组需要从性能、安全、稳定性等多方面进行评估。
* **配合测试与预发验证**：研发配合测试组在测试环境验证，通过后配合产品组在预发环境验证。
* **上线代码评审**：预发验证通过后，发起上线代码评审会议，由架构组进行二次检查。
* **上线checklist**：研发在上线前需要填写上线验证检查点，依次对需要留意的地方进行勾选。
* **上线**：遵守上线流程，规范上线流程。

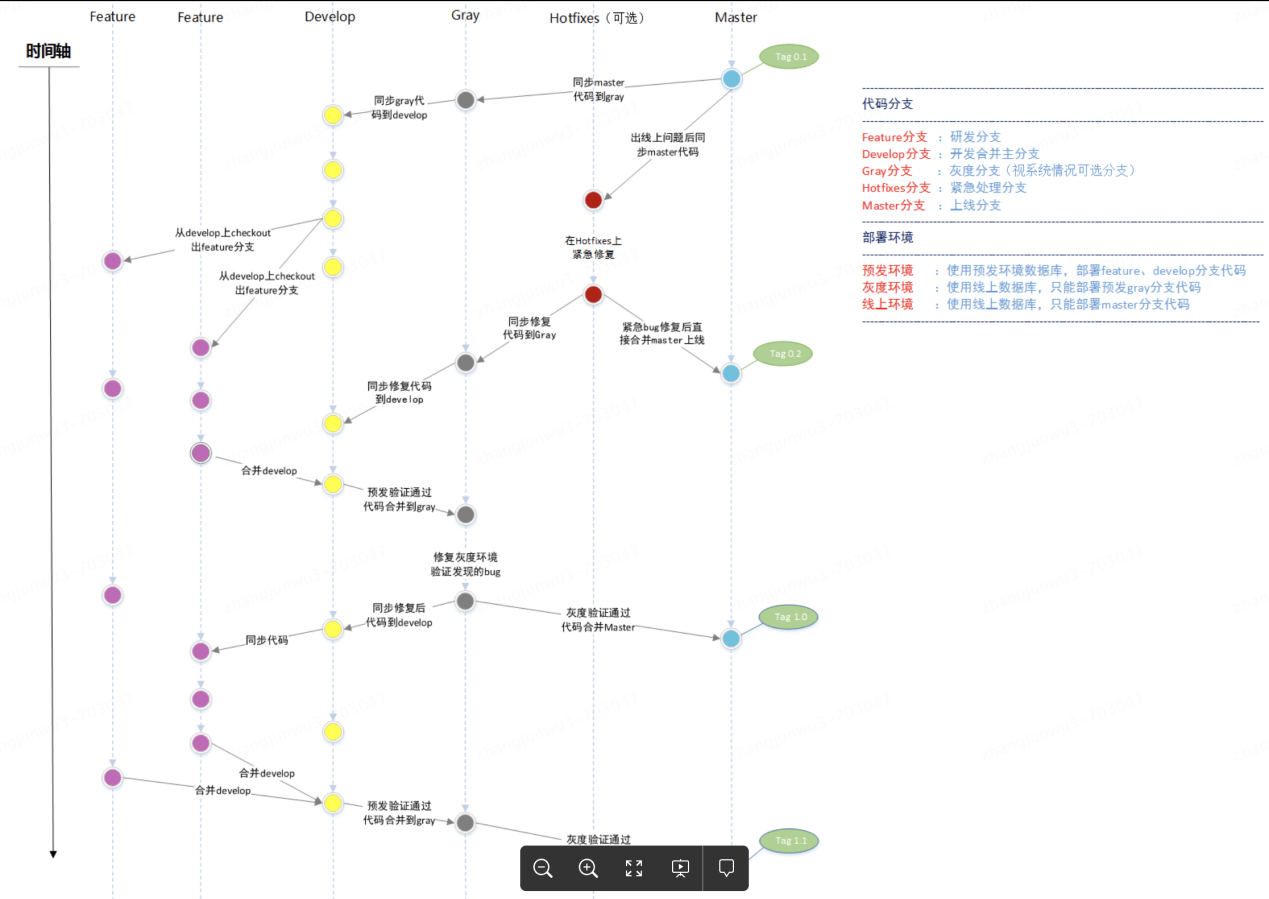
## 3.2 开发代码规范

这部分由于条目众多，会单独出一份代码规范文档，研发同学在开发代码过程中需要熟记，避免出现低等错误。



## 3.3 开发分支规范

良好的分支管理可以保障业务的开发迭代效能，并减少线上的事故率。



# 产品团队

## 4.1 产品职责分工



公司内业务分为多条业务线，每条业务线需要分配专门的产品进行跟进。具体细分需要根据当前业务与人员进行分配。专人专事可以提高效能，且提升与业务方的沟通能力。



## 4.2 知识库

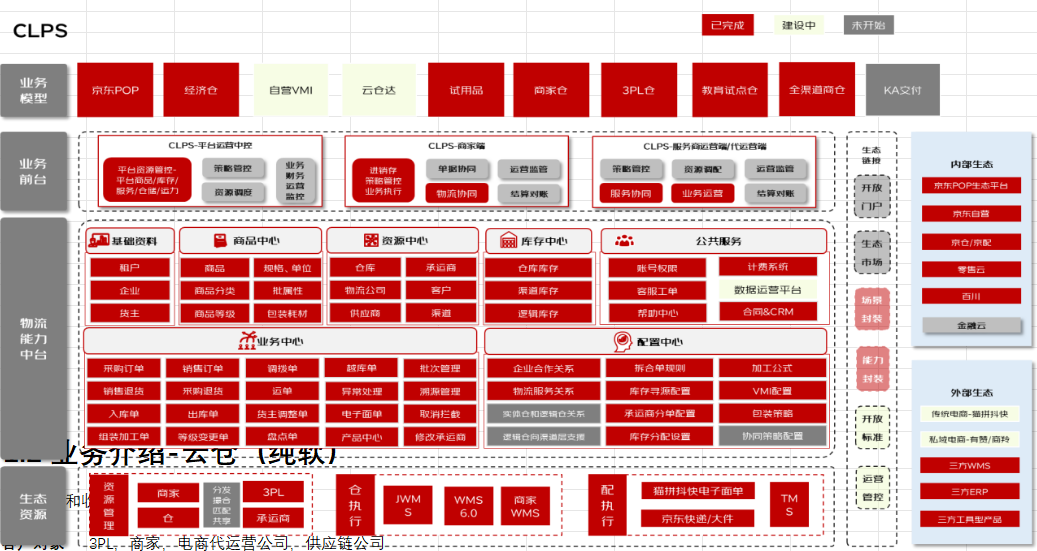


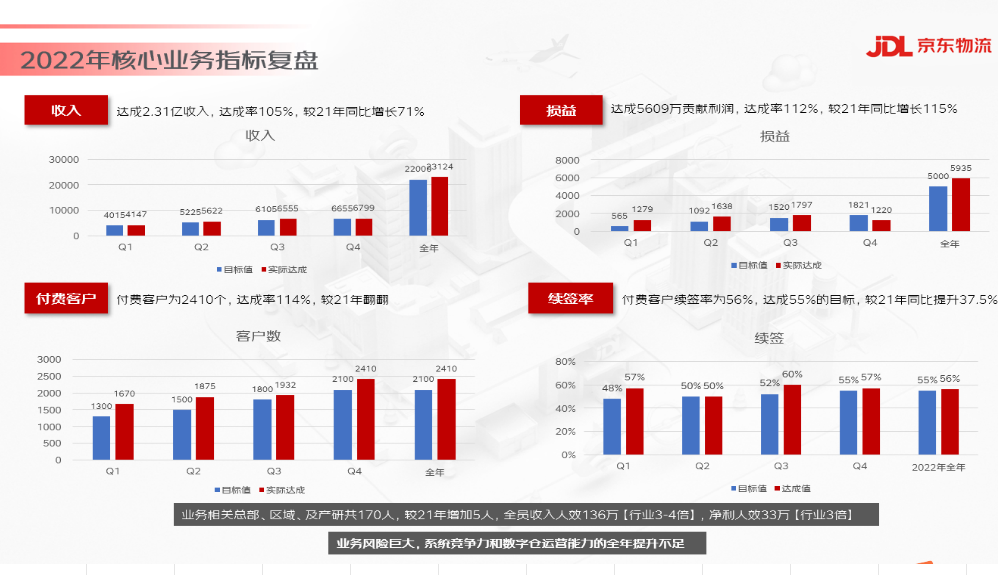
产品团队需要搭建自己的业务知识库，存放以往的BRD、PRD等资产，形成自己的产品能力。



## 4.3 业务全景图

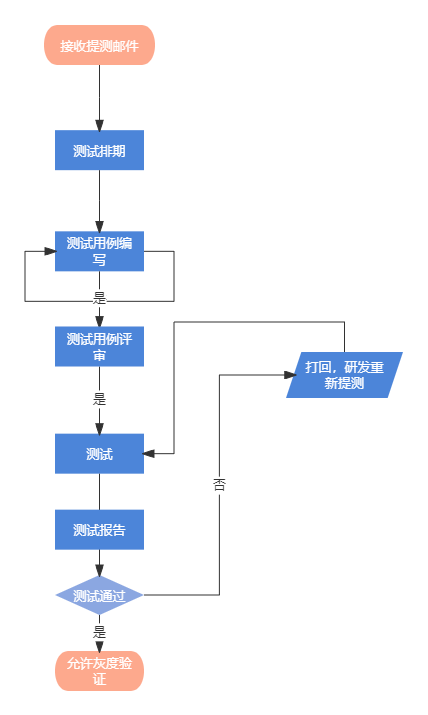
产品团队是业务方紧密沟通的门户，需要对业务方的需求进行理解、鉴定、收益对比分析等。产品需要对公司的整体业务有深刻的理解与认知，故搭建公司的业务全景图是非常有必要的。





# 测试团队

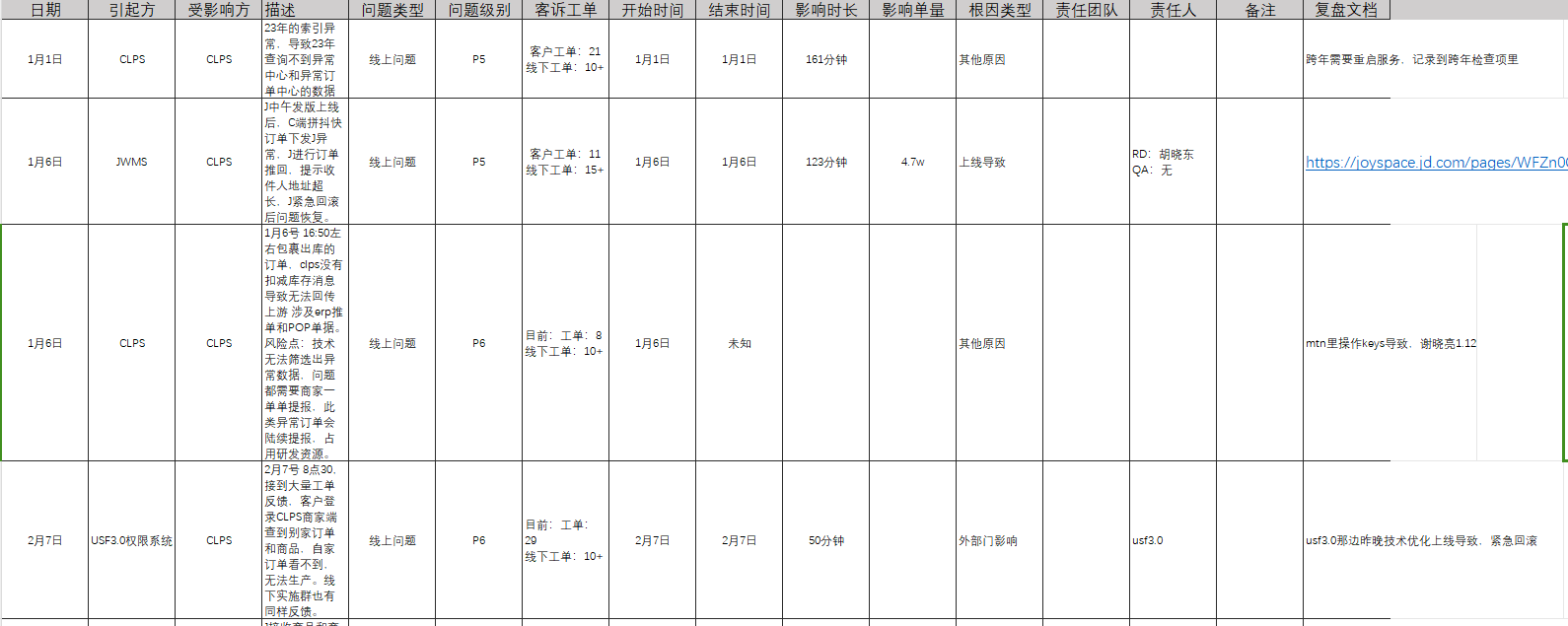
## 5.1 测试流程规范



* **接收提测邮件**：研发在自测后发出提测邮件给测试团队。
* **测试排期**：测试团队分配资源，给出工时排期。
* **测试用例编写**：测试人员接收需求后，需要编写功能用例文档。
* **测试用例评审**：测试人员编写文档后需要组织测试用例评审，测试与相应研发需要参与。
* **测试**：工期内正常测试，如延期，需要阐明延期原因并重新排期。
* **测试报告**：若测试通过，则出具测试报告，研发收到后联系产品开始预发环境验证。如未通过，则打回提测，阐明未通过原因，由研发人员修复后重新提测。

## 5.2 线上质量看板

线上质量看板用于统计在线平台出现的bug、事故率，分析出事故出现的具体原因，并分配到具体人员进行修复，它是一个逐渐优化的过程。



# 运维团队

## 6.1 平台手册维护

无论To B/C的产品，在新功能上线之后，运维团队都需要对平台使用手册进行更新，并对业务方或用户进行培训，提升服务质量。



## 6.2 工单维护

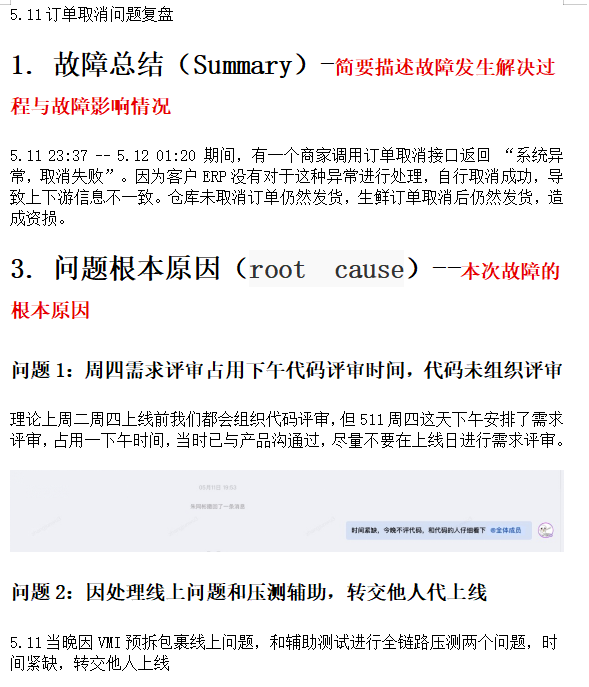
平台作为开放产品，需要开辟单独的工单板块，供用户对平台存在的问题进行提交反馈，可以提高用户的留存率。





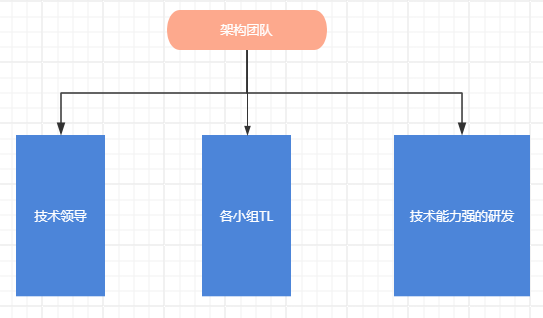
## 6.3线上事故复盘

线上事故不可避免，每次发生线上事故，运维团队是第一感知，需要及时通知研发团队查看分析原因，可以采取临时方案进行修复。但事后一定要进行复盘会议。复盘不是目的，目的是为了避免下次再发生类似事故。



# 架构团队

## 7.1 成员组成

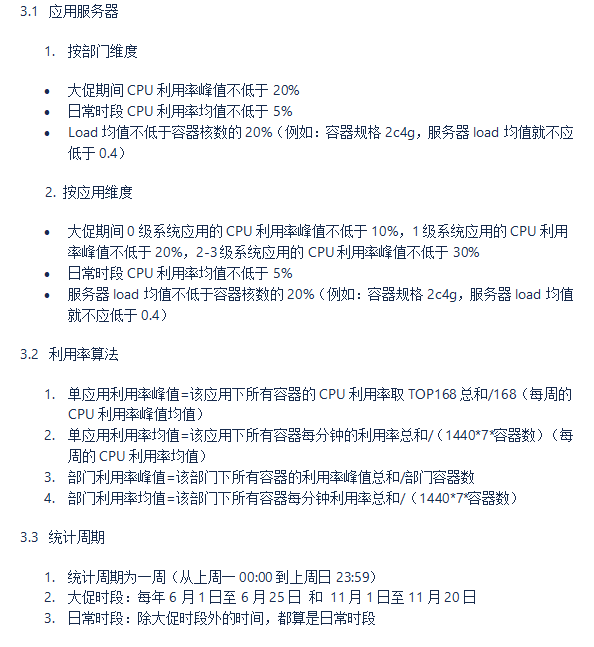


架构团队是虚线团队，其成员日常有着自己的工作职责。

## 7.2 规范制定

架构团队负责整个技术团队的规范制定，比如代码规范、开发分支规范、上线规范等。

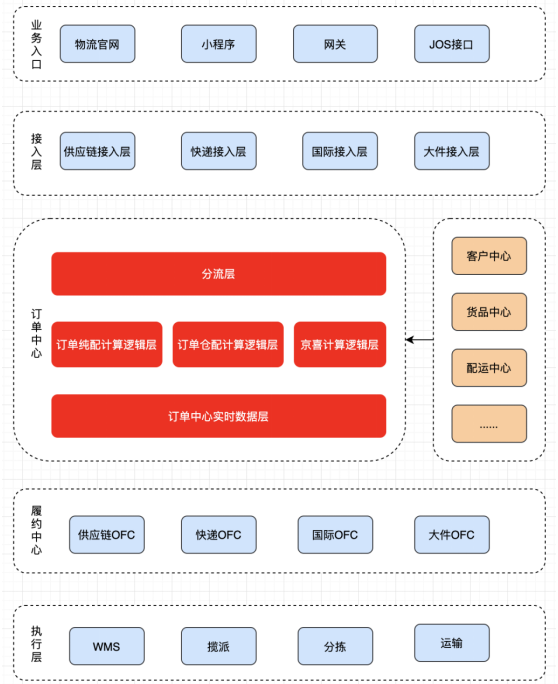
规范发布会有一定的公示期，公示期内技术团队所有人都可以提出自己的意见和想法，由架构团队评定后进行修改。公示期过后进行发布。一旦发布，团队所有人都需要遵守，这是硬性规定。



## 7.3 团队职责

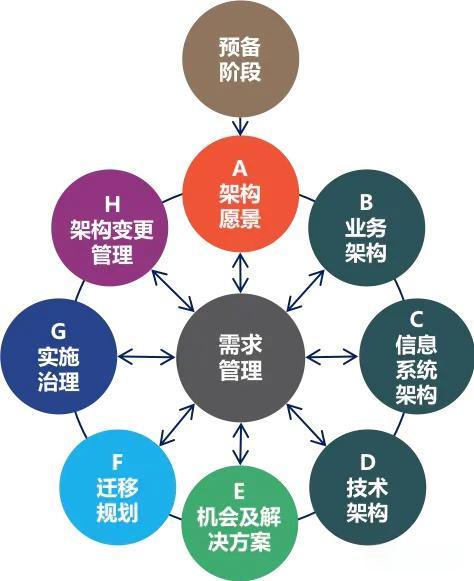
架构团队负责平台所有与架构相关的工作：

* 平台搭建之初的整体规划
* 业务迭代所需要的重构优化
* 框架与通用组件的开发等等
* 平台能力搭建
* Paas 中间件开发
* Iaas 基础设施



## 7.4 架构设计流程规范

企业架构方法有很多，但TOGAF是[最主流](https://baike.baidu.com/item/%E6%9C%80%E4%B8%BB%E6%B5%81/2753318?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/TOGAF/_blank)的。不仅有80%的福布斯( Forbes)全球排名前50的公司在使用，而且支持开放、标准的SOA参考架构。



TOGAF 架构开发方法 (ADM) 为开发架构提供了经过测试且可重复的过程。ADM 包括建立架构框架、开发架构内容、转换和管理架构的实现。

所有这些活动都在持续架构定义和实现的迭代循环中进行，允许组织以受控方式转变其企业以响应业务目标和机会。

### 阶段1：架构愿景

架构愿景描述了架构开发周期的初始阶段。它包括有关定义范围、识别利益相关者、创建架构愿景和获得批准的信息。

### 阶段2：业务架构

业务架构描述了业务架构的开发，以支持商定的架构愿景。

### 阶段 3：信息系统架构

描述了用于架构项目的信息系统架构的开发，包括数据和应用程序架构的开发。

### 阶段 4：技术架构

描述了架构项目的技术架构的开发。

### 阶段 5：机会和解决方案

进行初始实施规划和确定前几个阶段定义的架构的交付工具。

### 阶段 6：迁移规划

涉及制定一系列详细的过渡架构序列，并提供支持的实施和迁移计划。

### 阶段 7：实施治理

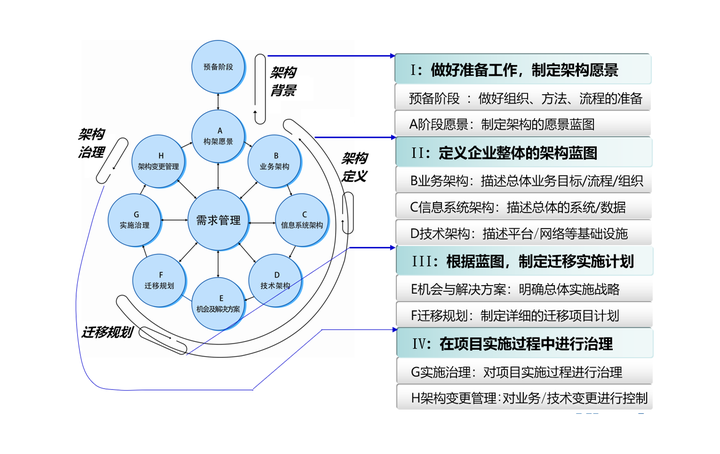
提供实施的架构监督。

### 阶段 8：架构变更管理

建立管理新架构变更的程序。

### 核心：需求管理检查

在整个 ADM 中管理架构需求的过程



## 7.5代码评审规范

研发人员在提测与上线前都需要申请进行代码评审，架构团队遵循代码审查规范，并且需要从性能、安全、稳定性等多方面进行考量，保障上线的稳定性。



# 总结

以上是个人对公司技术团队未来发展的一些粗略想法，真诚希望对各位公司的发展能起到一定作用，并且得到大家的评鉴，谢谢。

