知乎

浅谈银行业IT项目管理

唐唐的异想世界
3岁女宝的妈妈 996的金融从业者 日常所思所想

关注她

还记得刚入行之初，老行长对我们这些新任的项目经理是这么说的。

银行IT项目非常重要，某种程度上要比业务发展还要重要，现代银行是以IT为基础、为抓手，IT系统是业务发展的重要工具。IT项目的失败的定义不是没有成功上线，或者项目没有用，而是要看项目是否超时、上线时业务是否已经变化很大、业务无法使用。所以，我们在立项时要抱着谦卑的心态，认为项目一定是要失败的，我们要尽全力把它救起来。当年中行的项目管控非常严格，一个获了大奖的项目全球支付平台，从项目管理角度其实是一个失败的项目，最终让IBM多花了八千万。失败的原因在于立项时对后期的资源判断不足、架构管理的经验确实缺失、业务部门需求不清等。项目延期情况非常严重，整个市场都发生了变化，项目还没有上去。所以项目管理要把自己放在无以复加的境地，再去逐一想一想，需要什么资源和保障能让项目按期完成，这需要整个项目管理的方法论，包括如何管理厂商、内部协调等。

我自己也在老行长的指引下，踏上了项目经理的修行之路。从2013年第一次担任项目经理，当时的项目团队只有3个人，交付周期为4个月。到2020年负责新一代信贷系统的交付，项目团队有近30人，交付周期约18个月。在这个过程中增强了自己的业务知识，锻炼了自己的管理能力，所以也想对项目管理知识有个系统性的总结。

一、首先从流程制度的角度，来看项目管理

1、PMP项目管理流程

1）五大过程组

项目不是自动推进的，需要项目经理积极主动地推动，每个项目从开始到结束都会经历以下五个过程：

- **启动**：项目启动会议在这个阶段进行，领导开始画饼，同时任命项目经理。
- **规划**：虽然变化总比计划快，但是我们也不打无准备之仗。
- **执行**：管理和完成项目工作。
- **监控**：监控项目状态、发现项目问题、管理变更请求。
- **收尾**：项目验收、文件归档、总结经验教训。

2）十大知识领域

项目管理的核心思想是系统化思维，十大知识领域提供了全方位管理项目的思路，使你考虑问题更加全面。

- **相关方管理**：对能够影响项目的个人或组织进行管理，提高相关方对项目的支持力度。
- **沟通管理**：什么项目信息在哪些相关方之间以何种方式流动，以促进项目的实施。
- **风险管理**：管控项目风险和机会，提高项目的成功率。
- **范围管理**：明确工作内容，确保做且仅做项目需要的工作。拒绝范围蔓延和项目镀金。
- **进度管理**：先做什么、再做什么？按照项目范围和交付时间，规划并监控项目的全过程。
- **成本管理**：确保项目在批准的预算内完成。
- **质量管理**：确保项目产出满足相关方的质量要求。
- **资源管理**：不仅是实物资源，更要关注人力资源，团队的建设、管理以及士气的提升。
- **采购管理**：从组织外部采购获取项目所需资源。
- **整合管理**：项目经理的主要责任，贯穿项目始终，其重点是创建项目章程、制定项目管理计划、监控所有项目工作、管理变更。

关于十大知识领域，为方便理解和记忆笔者根据自己的工作经验串联如下：

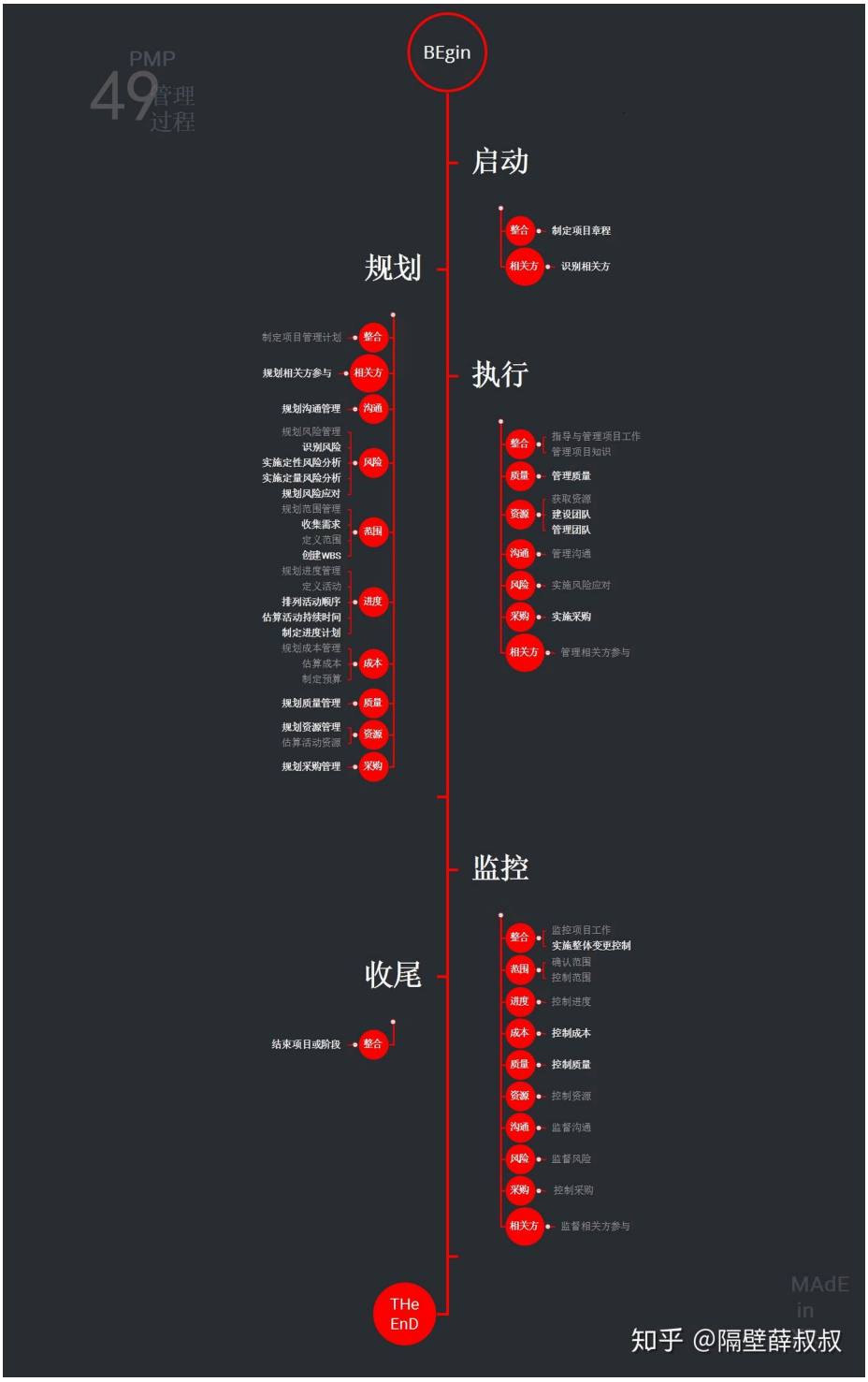
- 刚参加工作的时，往往对事不对人，这时候，项目在我们眼中是：**范围、进度、质量**

知乎

- 当你负责的项目越来越大，涉及的人和钱越来越多的时候，你不得不考虑：**资源、成本、风险，**资源不足时往往还需要进行**采购**

3）四十九个管理过程

启动、规划、执行、监控、收尾是每个项目必经的时间线，在这条时间线上，我们需要综合考虑十个知识领域。那么按照时间先后的顺序，我们在做项目的时候应该具体考虑哪些内容呢？四十九个管理子过程就回答了这个问题。我们首先通过下图，对四十九个管理过程做一个初步的认知



2、敏捷项目管理

在项目实践过程中，瀑布模型经常在以下 3 个方面饱受诟病：

1、研发周期过长，导致研发跟不上业务发展的节奏。

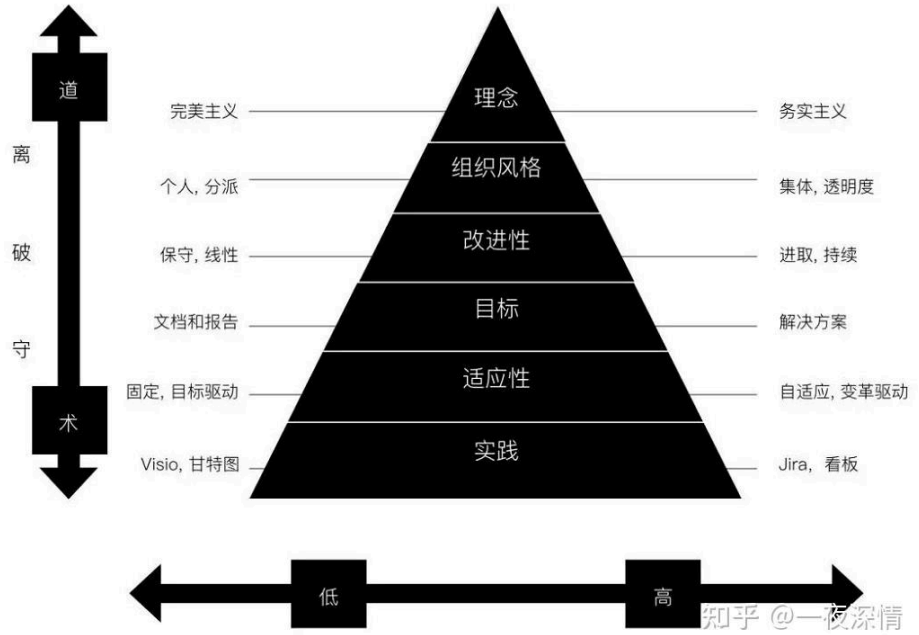
了，或者业务已经发生了很大变化。

2、研发不能很好地响应需求变化，导致客户满意度低。

要知道，我们很难在设计之初就把所有的需求都想清楚，而需求又是不断变化的，因此客户在看到真正的产品出来之前，对产品很可能是无感的，正如上面项目中的客户，他们在看到并试用了产品后才觉得这里应该这样，那里应该那样。瀑布模型是直到项目最后，才一次性地向客户交付产品，当产品被开发出来之后，客户才发现他们需要的不是这个东西，这是非常可怕的事情。

3、不能很好地管控风险。

这是因为研发到最后才一次性交付产品，所以项目中很多风险在前期很难被完全识别出来，那等到最后发现时再想处理，就要付出更大的代价了。



一、管理流程

完整的项目管理流程可以总结氛围五个过程组：启动、规划、执行、监控、收尾，敏捷项目管理框架是：构想、推测、探索、适应、结束，和PMBOK知识体系项目管理五大过程组一一对齐。

1、传统项目管理

传统的项目管理要对项目的所有过程进行管理和风险把控，并要求在不同环节有文档输入和输出。比如PMBOK对项目整合管理的过程组做了文档输入和输出的整理，但是，项目管理主要是对范围，进度、成本，质量、人力资源、沟通、风险、采购和干系人进行管理，每个环节都存在启动、规划、执行、监控和收尾的过程。

如果采用传统项目管理模式，每个环节都必须要进行严格的规划，一旦出现规划以外的变更，都需要经过批准后才能执行改变。

2、敏捷项目管理

敏捷项目管理简化了繁琐的流程和文档管理，主张团队内部面对面的沟通和交流，以Scrum为代表的的、简单、持续集成、不断交付、价值优先、拥抱变化的原则在面对时刻变化的市场经济和不断发展的技术时变得十分友好。敏捷项目中，项目管理计划分不同等级，可以用一个洋葱图来表示，也就是洋葱计划图，如下图：

战略和投资规划在敏捷项目管理的

知乎

最外层，由更广泛的组织管理系统来处理。由外往内，不断切分项目计划，最后实现最小周期的可行性版本迭代。对复杂或不明确的客户需求进行合理的分割，最终实现总体上的统一。

一句话，敏捷 = 价值观 + 原则 + 一系列符合价值观和原则的方法。单纯说敏捷是一种方法，肯定是片面的；但只强调它的价值观和原则，而不重视方法也是不对的，因为那样敏捷就飘在空中，不能落地了

二、Scrum的工作方法

Scrum 是一个敏捷项目管理理论框架，它通过多个固定周期的迭代（Sprint）开展工作，每个迭代都有四个主要会议。

迭代会从待办事项列表（Product Backlog）或目标任务开始。在迭代中，待办事项列表有两种：一种是产品待办事项列表（由产品负责人负责），这是产品功能的优先级列表；另一种是迭代待办事项列表，从产品待办事项列表获取目标任务安排到迭代，直至达到当前团队在一个迭代中能够完成的工作量上限。

在迭代中，Scrum 团队根据职责可以分为三种不同角色。通常为一名敏捷教练（Scrum Master），负责团队的敏捷开发指导；一名产品负责人（Product Owner），负责产品规划；以及开发团队，他们通常跨职能协作，负责迭代任务的执行和交付。

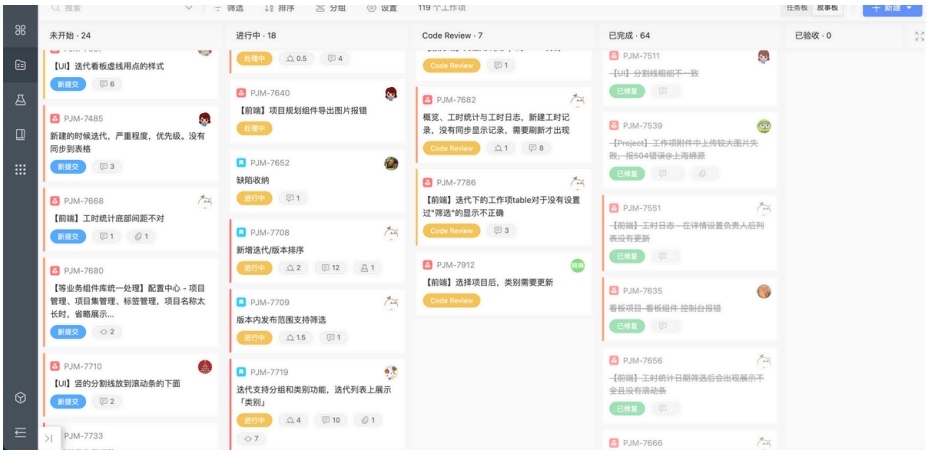
1、Scrum 四大会议

迭代计划会	每日站会	迭代评审会	迭代回顾会
一个团队计划会议，确定即将带来的迭代（Sprint）需要完成哪些任务。	也称为站立会议，是软件团队同步进行的 15 分钟小型会议，通常围绕三个问题展开。	一个共享会议，团队展示他们在该 Sprint 中交付的内容。	回顾哪些行动做得很好，哪些行动做得不好，使下一个 Sprint 变得更好。

2、Scrum 看板

Scrum 的迭代看板可以让团队一目了然地了解当前迭代所有任务的进度。在迭代计划会议中，研发团队会将目标任务从产品待办任务列表中移到迭代待办任务列表。在迭代看板中可以看到多个工作流状态，比如未开始、进行中和已完成等。Scrum 板是提高敏捷项目管理透明度的关键要素之一。

知乎



四、看板（Kanban）方法的工作方式

Kanban 是另一个敏捷项目管理的理论框架，它根据团队能力情况匹配工作量，着重于团队的快速交付，可以让研发团队做出比 Scrum 更迅速的变化响应。

与 Scrum 不同的是，Kanban 通常没有待办事项列表。任务一般安排在看板的"待办栏”中，随时可以执行和完成，以便 Kanban 团队能够专注于任务的持续交付。看板上的所有任务都可以让团队每位成员看到，而且具有规范的工作流程和进度；当某项任务完成后，研发人员可以马上就进行下一项任务。Kanban 通过在制品限制（WIP limits）来根据团队能力分配工作量，这是根据团队情况提前设置好的，对于每个看板栏里的可放置任务数量进行合理限制（待办栏没有限制）。

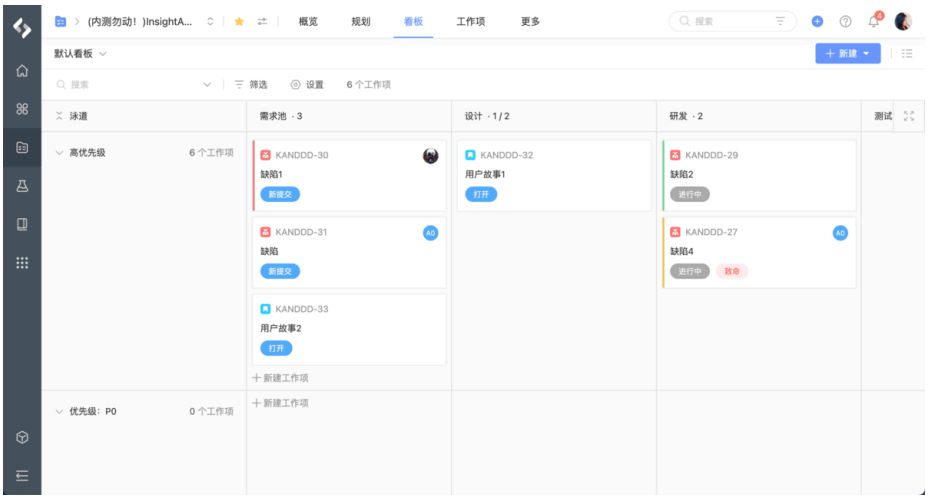
1、看板的四大组成部分

任务待办列表（用户故事列表）	看板栏/泳道	在制品限制（WIP Limit）	持续发布
指看板栏上需要完成的缺陷和任务。	用于区分任务的不同工作流状态、负责人、项目等。	指根据团队能力来限制每个看板栏的工作量。	该团队在 WIP 限制内处理故事数量，并且可以随时发布。

2、看板

看板可以让所有任务的工作进度一目了然。它还可以用来规划工作，帮助项目负责人根据团队情况制定计划。看板由代表着任务不同状态的看板栏组成，任务只有符合看板栏的在制品限制规则才能进入看板栏中，否则会继续被放置在待办栏里。Kanban 的待办栏任务会被划分为小颗粒度的任务，然后根据优先级来安排工作。

如示例图中的看板，通过“泳道”来区分高优先级和其他优先级的任务。



知乎

无论选择使用哪种敏捷方法进行研发项目的管理，都需要通过一种方式跟进团队的进展来进行长期规划以及迭代规划。敏捷项目估算可以让 Scrum 团队和 Kanban 团队了解自身工作能力；敏捷报表可以随时反映团队效率；待办事项列表可以让项目负责人保持工作列表的最新状态，以便团队随时能够处理。

1、敏捷项目估算

项目估算是 Kanban 和 Scrum 两种敏捷项目管理方式极为重要的一环。大多数的 Kanban 团队根据工作经验和团队规模来设置每个看板状态栏的在制品限制，而 Scrum 团队使用的是项目估算来确定每个迭代期间可以完成多少工作量。

很多敏捷开发的团队会采用特定的一些估算方式，如扑克牌估算、理想时间估算或故事点估算等来估算当前任务的所需工作量。项目估算为敏捷团队提供了参考，比如在迭代回顾会议上就可以对比了解团队的预期完成和实际完成任务的情况。

2、敏捷报表

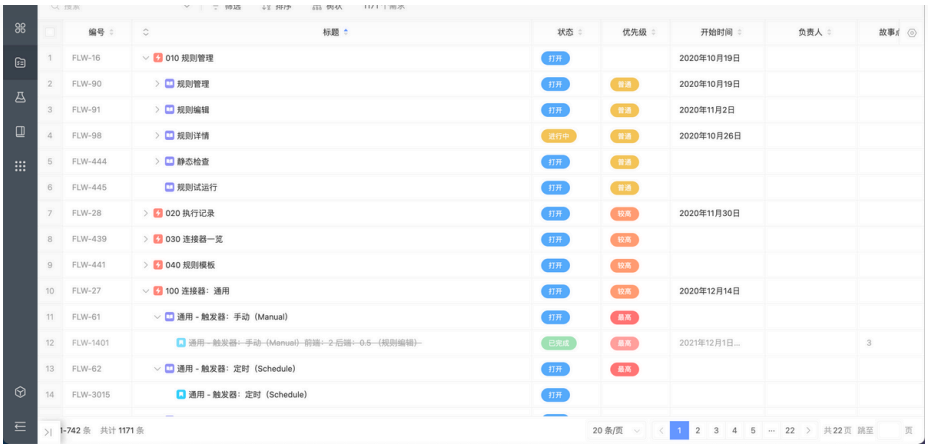
项目估算可以帮助研发团队在迭代开始时确定目标工作量，以及在迭代结束回顾时验证估算的准确性。但使用敏捷相关报表，例如燃尽图，可以帮助团队了解在迭代中完成了多少“故事点”。

比如，以下是工具 PingCode 提供的多种维度报表，如团队速度、成员负荷报等，支持查看实时数据，自定义配置维度，用数据复盘帮助敏捷团队快速迭代优化。

3、待办列表的管理

产品待办事项列表（ Product Backlog）是敏捷团队的工作优先级列表，列表的任务来源于产品路线图。研发团队的迭代任务都要从产品待办事项列表中获取，因此有效管理您的产品待办事项列表可以帮助团队达成更长远的目标，如根据团队情况不断调整迭代目标，以及根据市场需求调整业务目标等。

例如，我们自己就是通过 PingCode 的产品需求规划列表和迭代任务列表快速规划任务和跟踪任务进度，除此以外，还有新建任务、设置列表自定义表头展示字段、列表视图切换和排序、以及搜索和筛选任务等等可以使用。



3、银行业IT项目管理流程及管控点

1）银行业项目管理流程，主要分为启动阶段的立项流程和交付阶段的研发交付流程。其中立项流程包含高阶可研流程和详细可研流程。研发交付流程主要分为六大阶段，正式启动、需求分析、设计开发、测试、投产准备几投产、项目后评价

2）加强管控，包含预算管控、需求管控、质量管控

4、CMMI和ISO质量管理体系

5、IT项目管理的痛点和解决方案

痛点一：资源不足

讨论背景：
任何一家公司、任何一个部门永远不可能资源充足，管理人员要适应资源不足的常态，故而针对以下问题进行了内部讨论：

需求太多，产品啥都要?
工期紧张，立flag卡时间?
人员有限，有hc招不到人?
依赖太多，状况百出，各种延期?

交流心得：

一、需求太多--排优先级

1.1.2. C端用户体验优先
1.1.3. 探讨方案替代(看透需求本质，产品要的是过河，不一定是船/桥)
1.1.4. 产品妥协、降级(先上线，再迭代)
1.1.5. prd需求质量(同频共振，产研理解一致)

1.2. 技术
1.2.1. 复用已有组件/服务
1.2.2. 代码规范性，风格可适当共存
1.2.3. 扩展性(预留2期,3期)
1.2.4. 架构不过度设计
1.2.5. 合二为一(技术类优化可和产品需求合并，一次性开发、测试、上线)
1.2.6. 拔插式(部分服务没改造之前，替代服务先使用，后局部升级)

二、工期紧张--提升效率
2.1. 架构方案折中(没有完美架构，先实现需求，再迭代优化)
2.2. 人员分工(经验丰富更熟练人员做最快)
2.3. 前紧后松(工期往前赶，余留buffer)
2.4. 集中封闭
2.5. 分模块并行提测
2.6. 任务拆解详细(粒度越细，把控更精准)
2.7. 外部依赖协调(高优解决外部，明确需要对方做什么，时间点)
2.8. 同步做(服务、权限、数据库，开发和线上一起申请)
2.9. 做预案 plan B
2.10. 整理上线todoList
2.11. 人员借调支援
2.12. 定时check（站会，晨会，周会）
2.13. 求教高人(技术难点，多问多沟通)
2.14. 当面求教(不限于IM沟通，电话，当面)

三、招不到人--注重平时积累
3.1. 靠介绍(主动找朋友、同事、前同事)
3.2. 自己去平台找(BOSS，拉勾、脉脉)
3.3. 广告(朋友圈、社交平台宣传)
3.4. 外部影响力(公众号、社区、技术答疑引流)
3.5. 推荐奖励
3.6. 提升现有人员能力
3.7. 提升研发效率
3.8. 岗位核心竞争力、岗位亮点
3.9. 适当放宽门槛
3.10. 适当提升福利待遇
3.11. 人脉积累

四、依赖太多--学会适应别人
4.1. 提前节点沟通(立项、开发、联调、测试、上线)
4.2. 主动沟通(必要可升级沟通方式：电话，当面)
4.3. 做预案 plan B
4.4. 风险预警，问题向上升级
4.5. 维护人际关系
4.6. 站在别人角度想(解决他的困扰，eg：陪加班可开车送回家等)
4.7. 上线后感谢信
4.8. 请老板站台
4.9. 自己人(不同部门都有处的好的朋友，通过朋友再推动)
4.10. 先看文档，带着问题结论寻求帮助(服务入参、出参等)
综上，核心三要素：MVP迭代，提升效率/能力，注重平时积累。

痛点二：情绪牵制

知乎

项目经理一个很重要的职能就是沟通协调，多数项目管理人员缺少风险预判能力，当出现面临风险和出现问题时又不能很好地向项目各方反馈，更提不出好的解决方案。比如多数技术转型项目管理人员面对不合理需求时，不敢也不会向客户反馈、更不会向公司领导表述，任务强压给团队成员后，又不能安抚好员工情绪，结果可想而知。对于不懂技术项目管理人员的问题是分析问题（尤其是技术问题）不深入，往往口头禅是这块儿不太懂、这块儿没想到或没想明白，由XXX来说，开个会恨不得能拉上整个团队。回头分配任务，要不就是发号施令，要不就是传声筒，从而逐渐丧失威信。

项目经理作为一线基础管理人员，对外有客户、甲方、用户、监理，对上有领导，对内项目团队成员，需要合理规划、统筹安排、有效沟通才能真正把项目做好。

交流心得：

一、领导不满意

1、信息传递失真
1.1、多讨论
1.2、反讲确认
1.3、快速反馈

2、团队效率不高

3、风险识别不够
3.1、传授方法论
3.2、亲自带着做
3.3、了解上下游
3.4、了解每个人困扰
3.5、多深入一线

4、主动向上汇报

5、主动性热情不够
5.1、权利不够
5.2、价值没说清楚
5.3、引导赞美鼓励
5.4、提升视野、认知
5.5、关键先生
5.6、关系维护

6、下属问题麻烦制造者
6.1、润物细无声，不是针对他，所有人都一样
6.2、配备back up，弥补他
6.3、规范流程、明确制度要求
6.4、能力差—提升技术能力
6.5、粗心大意—责令改进
6.6、陪他一起习惯培养
6.7、准备工作做充分

7、军令状flag变来变去

知乎

7.3、沟通不够
7.4、粒度粗，不够细致
7.5、转危为机，找更优解

8、描绘很好，结果没执行到位—不切实际给老板吹

9、老板基本工作要求未达成—定期周知老板，及时矫正

10、有担当，不找借口

二、员工闹情绪

1、工作分配不合理
1.1、了解每个人职业诉求，尽量满足
1.2、是否愿意
1.3、工作量太多
1.4、长期固定

2、公众场合被批评—私底下沟通

3、被甩锅、委屈
3.1、帮他澄清
3.2、边界划清楚
3.3、适当安抚
3.4、风险提前周知(事实)
3.5、对结果保证(先于别人之前发现问题)

4、边缘业务，没价值没成长
4.1、轮换制度
4.2、提高要求
4.3、技术赋能
4.4、关停下线(说服大家)
4.5、真诚沟通
4.6、放任务池，等主动认领

5、做得越多、错越多

6、队友不给力
6.1、摆正心态
6.2、互助提高
6.3、宽容、别较真

7、待遇被倒挂
7.1、所做事情要证明价值
7.2、成长性，放权给他
7.3、主动给予

8、荣誉奖励不公

知乎

8.3、透明、公信力
8.4、其它方面补偿
8.5、给最需要的人

9、性格孤僻、与团队不和

10、被领导画大饼

10.1、不要期待太多
10.2、领导尽量言行一致

11、被领导PUA
11.1、量力而行
11.2、容忍性
11.3、听一听就行

12、情绪低落
12.1、团队关怀
12.2、自我调节
12.3、情绪宣泄
12.4、团队荣誉感
综上，面对项目团队负面情绪，作为项目经理要及时感知并调整。最重要的是：把事情做正确。

痛点三：全局迷失

关键词：没有全局规划能力，资源调配失衡，目标不清晰

比较典型是水来土掩兵来将挡型，一事一议，全然没有计划型，结果导致资源调配失衡，不是资源浪费就是资源短缺。往往需求来了，不加以深入分析和设计，就急急忙忙向公司申请一大批开发人员介入，后期又过早把人员释放，导致测试阶段和上线后问题不断，没有人员及时修复系统缺陷。

项目管理人员从接触需求开始，就要全面分析需求、任务拆解、阶段划分、人员配置等一系列工作，建立项目整体计划并按计划推进和资源调配。

交流心得：

一、为什么要分期

1.为什么分期
1.1 项目定位
1.1.1 创新试点类型--最小MVP，敏捷迭代
1.1.2 稳定正常类型--固定跟版制、班车制
1.1.3 专项聚焦类型--集中资源，快速完结

1.2 资源短缺被迫拆解
1.3 外部依赖不可控
1.4 迭代开发，目标清晰
1.5 阶段性验证结果
1.6 规模小，可控制

知乎

二、没有规划

2.1 紧急需求插入，节奏被打乱
2.1.1 尽量减少需求插入--提前预知，未来2个版本要做啥
2.1.2 留buffer--积少成多，会导致大时间轴被拉长
2.1.3 政策不可控--长期关注行业政策，提前准备
2.1.4 线上业务逻辑不合理
2.1.5 体验优化类
2.1.6 老板需求--敢于挑战老板(虽然最后还是乖乖做)

2.2 人员变动
2.2.1 留出熟系/交接时间
2.2.2 平时培养backup
2.2.3 文档沉淀--前人栽树后人乘凉，衡量文档写得好不好，看第二个人能不能比第一个人时间更短，更快。
2.2.4 局部模块明确--把业务需求翻译为技术需求，直接开发

2.3 需求不明确
2.3.1 目标不够明确
2.3.2 战略决策调整
2.3.3 逻辑不明确--不同产品顾此失彼，缺乏全盘熟悉，整个项目团队都梳理维护，技术补位
2.3.4 测试用例，分支不够全
2.3.5 边界不清晰
2.3.6 理解不一致，没达成共识
2.3.7 对上下游了解不够

三、资源调配
3.1 项目拆解粒度不够细
3.2 需求理解不一致，不透彻--没考虑问题根本，而是表象
3.3 调研不够充分
3.4 联调阶段，阻塞block
3.5 外部依赖风险评估--可参考已接入案例，上期迭代等
3.6 大量工单走流程、审批--提前了解，提前申请
3.7 永远要有plain B
3.8 对项目里每个人能力有准确了解
3.9 人员规模过大，沟通成本--项目团队规模控制10人以内闭环。

痛点四：角色固化

关键词：忙于细节，忽视角色变化：从拉马车到赶马车

从技术岗转型到管理岗的项目经理人员通病，多数就是因为技术和业务能力比较强，被领导重视和认可，逐渐过渡到带团队，进而转型到项目管理工作。可是角色没有及时调整和转换，更多精力在于技术和业务细节上，忽视了整个项目的协调工作。项目成员不能放手去做，依赖性强，项目经理本身还有一堆文档要写、协调事儿要处理示，结果就是项目经理四处救火，忙得要死，且团队成员也很难独撑一面。

项目经理的作用：正确定义目标、制定项目计划、推动执行、进行团队建设及跨部门协调、保证质量、保证沟通、控制成本、密切监控过程并纠偏等，这样才能保证项目最终的成功。

知乎

一、去中心化

1.去中心化
1.1 提前全局规划、分配
1.2. 所有人对项目全局了解(这期功能点，目前进度，哪天测试，哪天上线)
1.3. 提前培养 backup
1.4. 文档沉淀，“错题本”积累
1.5. 责任边界，所有人清晰谁负责哪些模块
1.6. 敢于充分授权，只负责跟进
1.7. 同角色内部多分享，所有人都快速接手
1.8. 项目组内部信息广播，传递(开大会，全员周知，不限于微信群，邮件)

二、流程规范

2.1. 之前做得好的，列出来借鉴
2.2. 兼职pmo，不能只关注自己开发任务
2.3. 整理项目管理清单，每日一问
2.4. 随时把控关键角色
2.5. 关键节点check，有明确各方配合方案
2.6. 达成共识，不同阶段该做什么
2.7. 至少组织3次全员会，需求评审，排期，联调提测

三、到处救火

3.1. 提前评估难度、风险，并做准备
3.2. 子模块业务可以独当一面
3.3. 正确的行动计划(重要&紧急)
3.4. 留出buffer时间
3.5. 做好复盘
3.6. 通过之前案例找到解决方案
3.7. 做事方法论的沉淀
3.8. 提升大家解决问题思路和能力
3.9. 做些通用demo可参考，只负责难点攻坚

痛点五：利益分配

关键词：部门不同，目标不同，利益不同，利益分配的合理性

人们奋斗所争取的一切，都同他们的利益有关，这是马克思的至理名言。人们为团队工作，总要获得利益，或物质的，或精神的。利益的分配，代表着一个人的贡献和成就。必须公平合理，同工同酬，论功行赏，这样才可以调动职工的积极性，提高团队士气；反之，就会引起职工的不满，挫伤职工的积极性，降低团队的士气。

交流心得：

一、多劳多得

(前提是要让所有人认可规则，否则依然不公平)
1.1 项目角色(核心参与者)
1.2 简单配合支持方要感谢(上线邮件、复盘会、当他领导面感谢)
1.3 目标把控者
1.4 过程中表现(平时收集细节)
1.5 项目攻关人物

二、核心价值

2.3 项目实际收益
2.4 独有性(为什么是我们做)
2.5 关键先生，较大突破

三、平衡未来
3.1 动态看不同角色权重
3.2 提前考虑未来依赖部门
3.3 平时注重挖掘潜力人员

四、分配不合理(负面)
4.1 不愿意配合(外部门)
4.2 消极、应付(项目内成员)
4.3 onwer威望值下降
4.4 得不到认同感，人才流失
4.5 提前与各方领导申请资源，方便协调
上面列举现象中，大部分之前已经给过解决答案，或者问题本来就很简单，不需要展开讨论。综上，希望能对大家在项目管理过程中有所启发。

二、接下来从人的角度看项目管理的组织架构职业发展路径

1、PM和PMO的差异

要搞清楚，PMO不是单纯的PM，为了完成项目交付而存在的。

如果只是这个目的，有项目经理就够了。PMO的存在，是为了让负责交付项目的项目经理更加优秀。让项目经理知道什么是好的标准，要怎么做?

PMO是很多项目经理职业生涯规划中的重要一环，这个职位要求我们理解整个组织结构，并且学会用战略目光审视项目与管理。

经常和一些朋友聊到PMO的未来发展，有些人持消极态度，因为大多数人发展自己坐着支持型的工作初级是可以比较容易晋升和成长，但中后期往往遇到天花板。

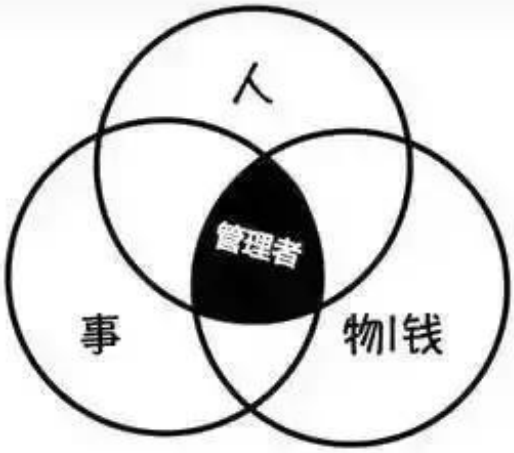
但其实我觉得**天花板还是要落到能力本身**，PMO是个特殊的角色，身上带着经验攒出来的管理能力。**最重要的根本就是不断提升自己，提升能力**

2、项目经理的职业发展路径

项目经理也有三六九等？初级项目经理和高级项目经理是有很大区别的，看看下面的介绍就知道你是哪种类型的项目经理了。

项目经理作为一名管理者，管理的内容繁多，成为一名优秀的管理者真心不易！他们要管人、理事，还要保证利润！不同的管理理念和水平直接制约着管理的效果。

知乎



初级项目经理与高级项目经理主要差距有以下几个方面：

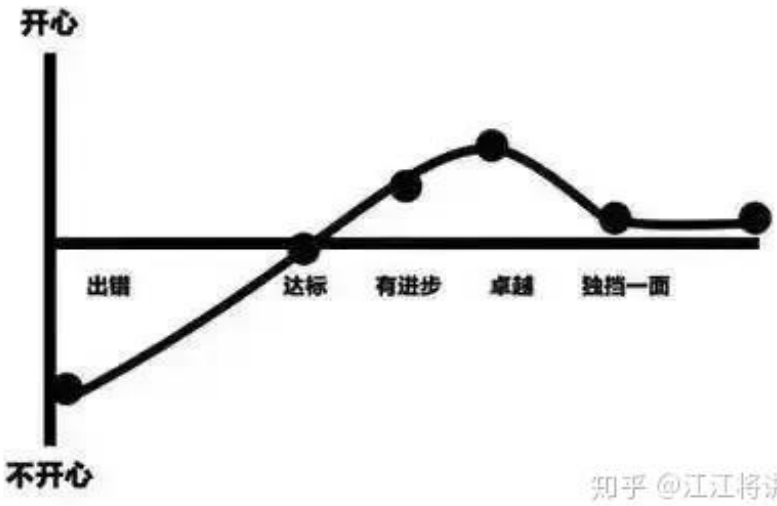
01

对下属的态度

情商是管理者重要的特质，高情商在管理者中主要表现为情绪稳定。

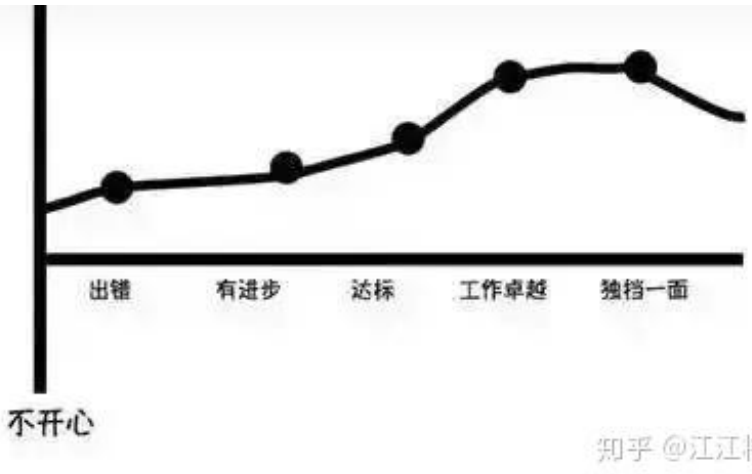
1. 初级项目经理

对于初级项目经理而言，他们的情绪很容易被员工的行为左右，并不能及时地调整，员工一旦出错或者是工作表现不理想，初级项目经理就会陷入到负面的情绪中。



2.高级项目经理

高级项目经理多数都有的一个共同特点是他们有很高的能量状态：你与他们交谈或共事的时候，感觉会非常舒服，也会令人敬佩。他们本身就是一枚暖男或者是女神，温文尔雅，给你的印象是阳光、自信；如果你一定要持反对意见，他或许已经走到了高级项目经理的职位，却依然是初级项目经理的心态。



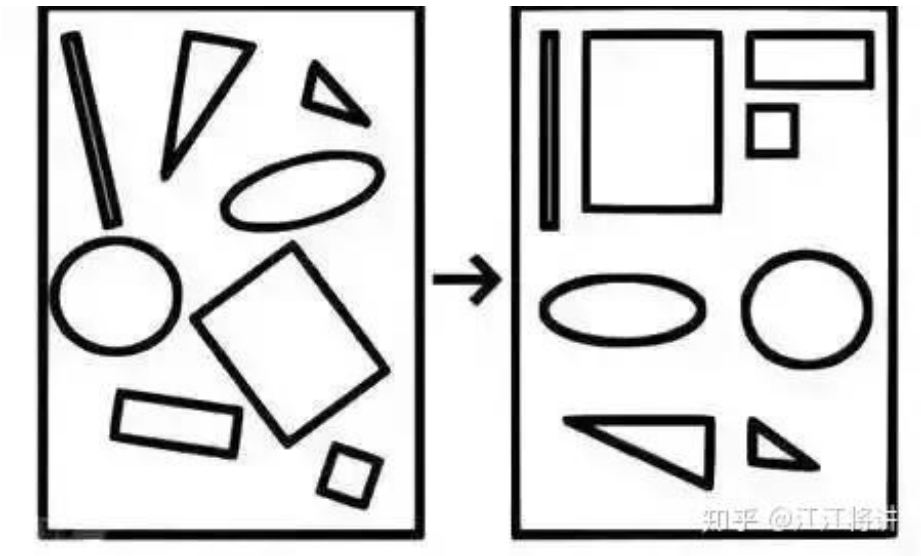
02

思维方式与工作追求

1.初级项目经理

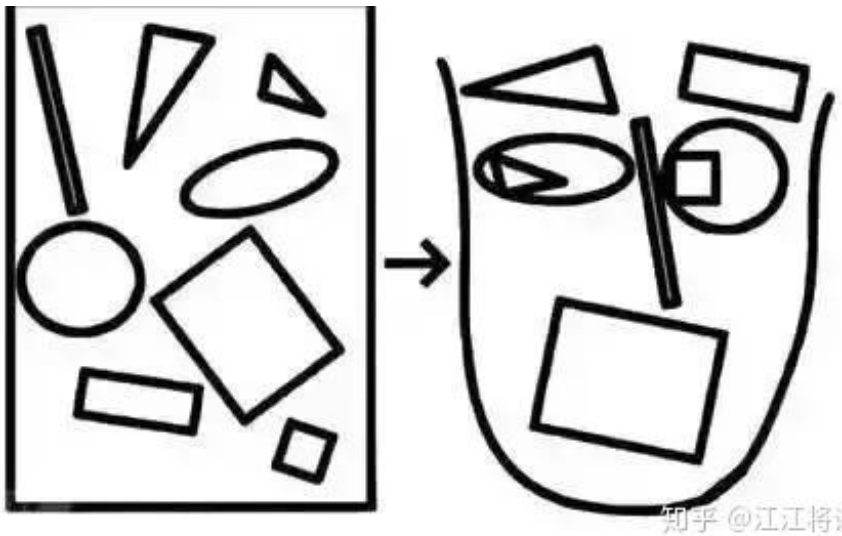
思维方式：初级项目经理能够意识到梳理业务流程的重要性及意义，但是由于缺乏基础的逻辑，不能够理解业务模块之间的内在逻辑，还是处于简单拼凑工作的状态，人、事、物的综合利用率较低。

表达方式：这张图的另外一层含义也在于管理者向团队成员传递企业文化、顾客要求及任务目标中出现的情况，一句话：茶壶里煮饺子，有话说不出！



2. 高级项目经理

思维方式：高级项目经理能够理解业务模块之间的内在逻辑，将团队内部的人、事、物进行综合利用，尽可能做到人尽其才、物尽其用，能够将每个要素安排到一个合适的位置，在他的思维里没有废料及边角料的概念。



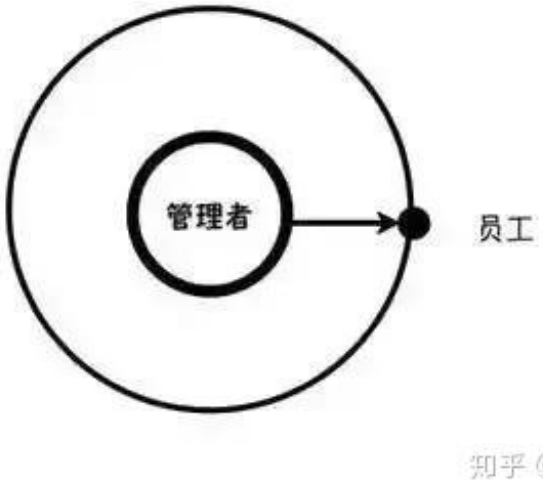
03

面对工作失误的反应

1. 初级项目经理

初级项目经理面对工作失误的时候，往往会先看到员工的问题，认为导致错误结果的原因都是员工执行不力，并粗暴地责怪员工，而忽略了自己在以下方面应该承担的管理职责：

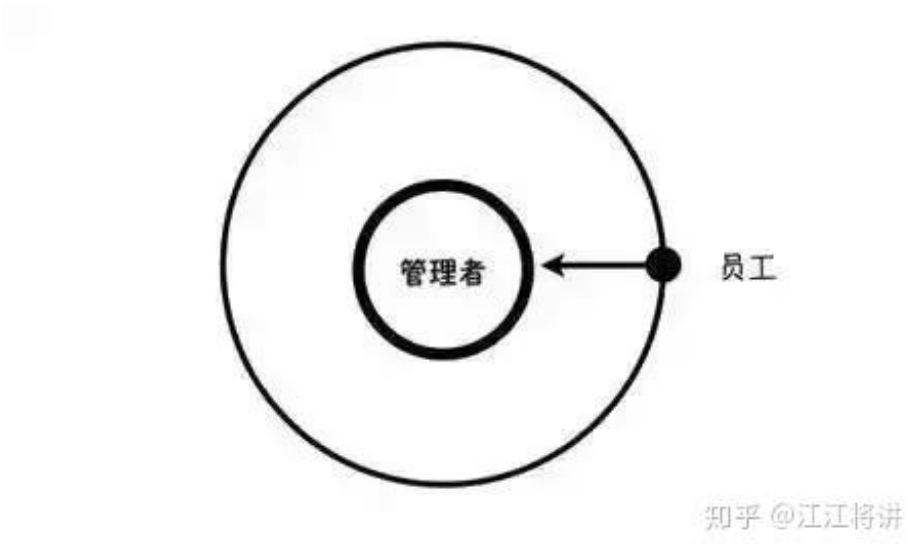
- 是否选对人了？
- 任务是否都讲清楚了？
- 有对员工进行辅导吗？
- 有激励员工吗？
- 资源配置够吗？
- 其他部门合作吗？



2. 高级项目经理

知乎

解决问题！

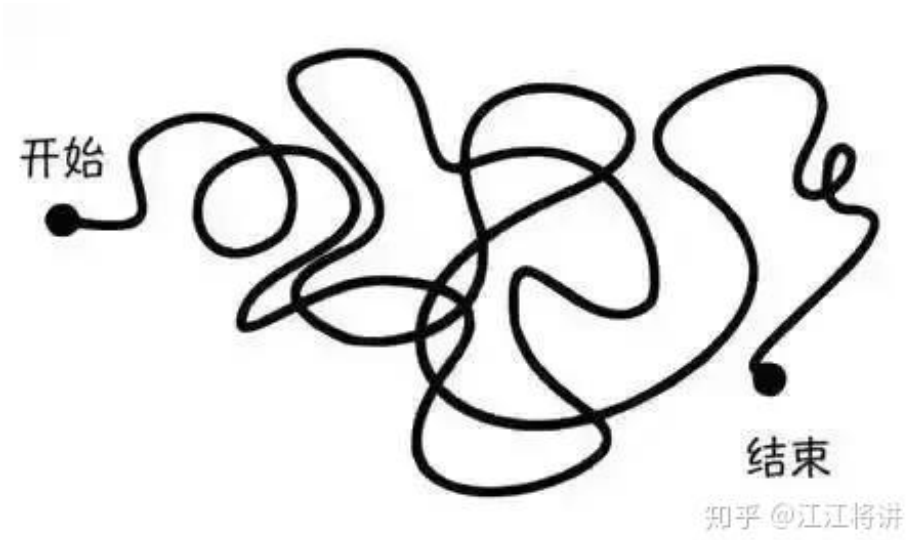


04

工作流程

1. 初级项目经理

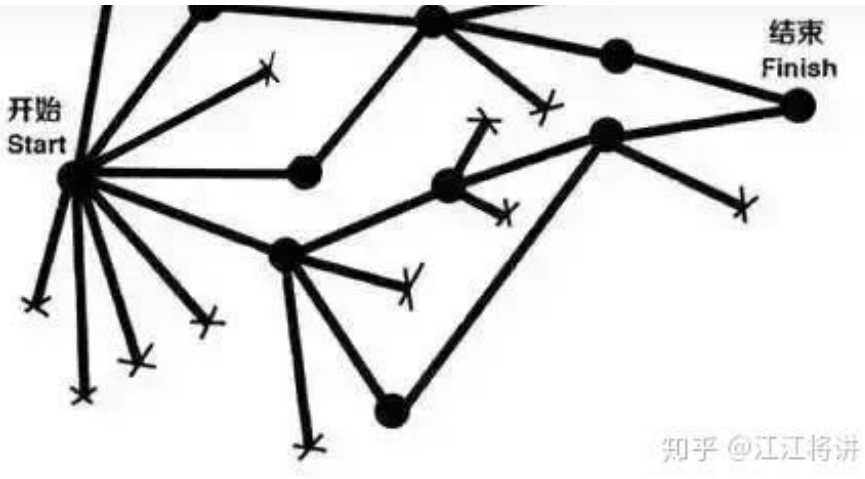
初级项目经理的一个典型问题就是对团队工作缺乏规划，很多工作按照自己的逻辑去开展，而忽略了团队式的工作方法，在实际工作中最典型的例子就是临时工作任务多、突发任务多，整个团队都会被带入到这种恶性循环，管理者及团队成员都会身心疲惫。



2. 高级项目经理

高级项目经理典型的工作方法就是对部门工作进行项目化及模块化管理，并能够基于部门或公司的绩效要求清晰地规划部门工作路径，能够知道在什么时候、团队中的每个人应该做什么事情，并且能够不断地优化这条路径，让部门运作有条不紊！

知乎

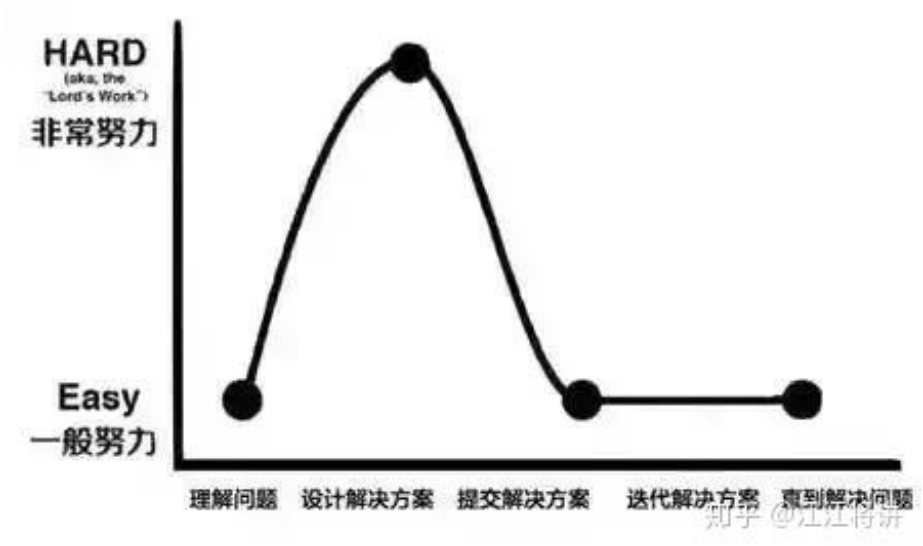


05

工作状态

1. 初级项目经理

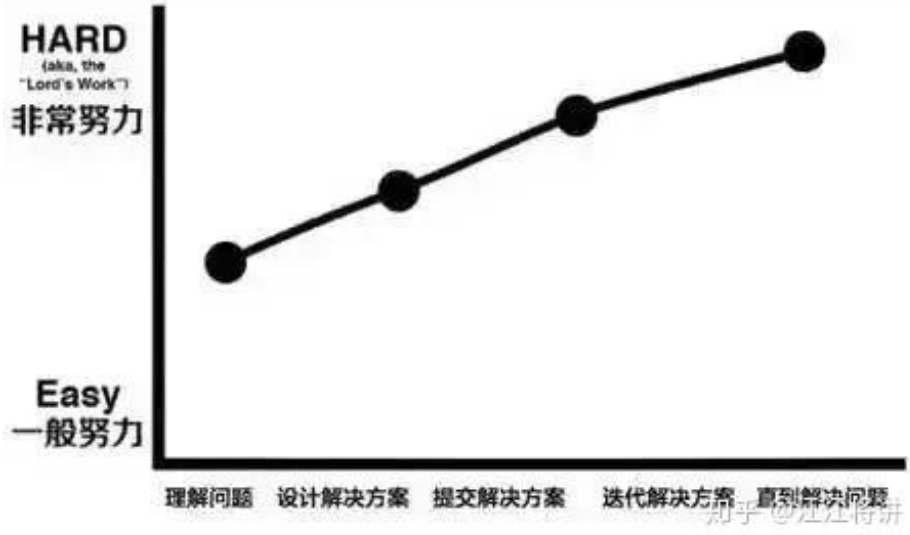
很多初级项目经理存在一种状态，我们将这种状态称为“三分钟热血”，即在理解任务阶段，缺乏耐心与细心，不愿意聆听顾客与上司的意见，而急于动手去做；在设计解决方案阶段则是天马行空，想象力很丰富，目标很宏大；在执行阶段却一下子蔫下去了；这是我们讲的眼高手低，也是不能够踏实做事的表现。



2. 高级项目经理

高级项目经理的工作状态就是一直保持着较高的激情。在他们的思维当中，已经形成了一种自我激励的机制，这种机制就是在解决问题的过程中获得乐趣，能够坚持到最后，见证自己的想法得以实现。从而达到了马斯洛所讲的“自我实现”的阶段。从这方面看，高级项目经理从来都不是为了钱而工作，却在工作中赚到了足够的钱。

知乎

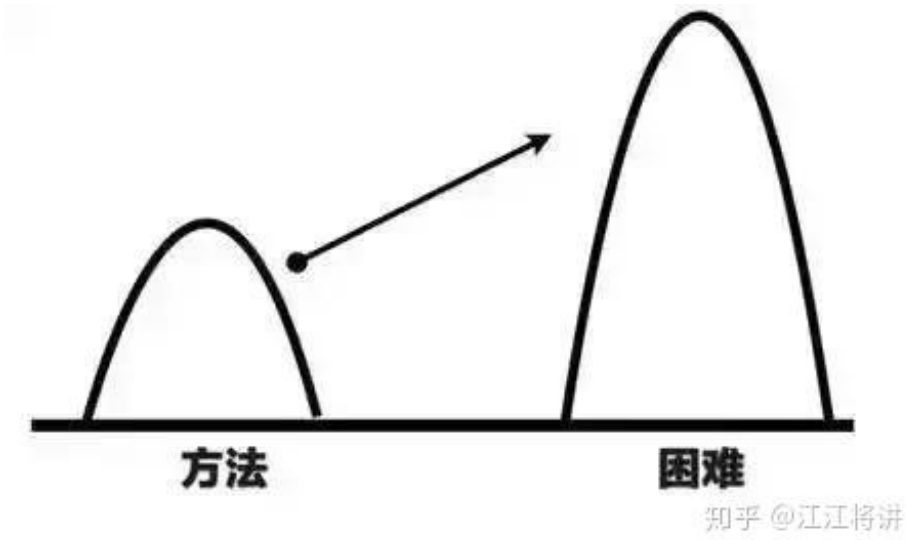


06

面对困难的心态

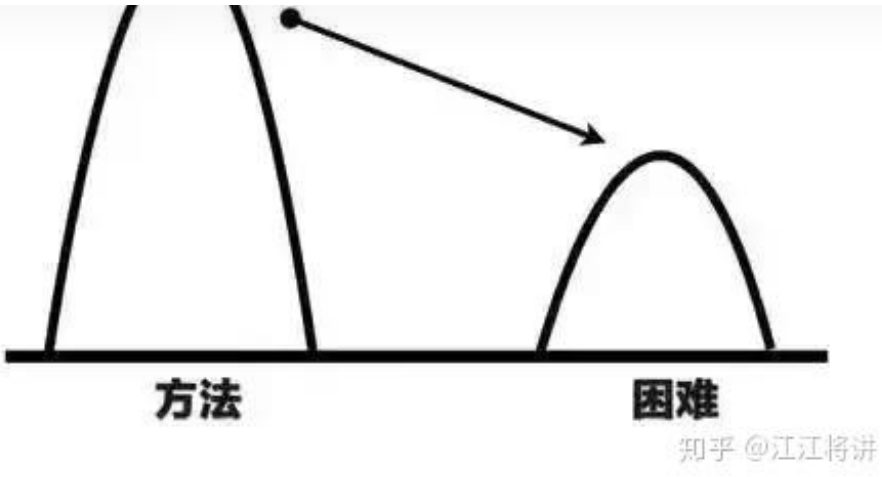
1. 初级项目经理

初级项目经理面对问题的角度总是站在山脚下，看到的往往是困难、挫败或风险，对于团队传递的往往是抱怨或不满。



2. 高级项目经理

高级项目经理对待问题的角度却是站在山头看山脚，能够看到解决问题的无限可能，将隐藏在问题背后的方法、收益挖掘出来，并带领团队去享受挖掘的过程与丰厚收益，成为商战中最大的赢家！



如何成为高级项目经理

1、管理时间就是管理自己，高效利用时间。

2、分清各项工作的轻重缓急。

3、不断规范和调整制度，没有规矩不成方圆。

4、提高会议效率，事前告诉大家会议的内容。

5、统计数据，针对数据进行分析，分析结果加以应用，最后不忘评估、验证成果。

6、愿景引来注意，尊重加深信心，沟通加强意义，立场导致信任。

7、定目标，严格执行、考核、监督。

8、人不要在意会什么，关键在于你会学什么。

9、培养人才资产：选、养、育、用、留。

10、成绩好的时候要考虑如何提高团队的建设

3、项目经理到底要不要懂技术?

不懂技术的PM，最终会尝尽生活的苦

01）项目经理可以不懂技术，但不能不懂业务

不懂技术的PM也不要担心，项目经理圈子中有许多非技术出身的管理者，也非常优秀。

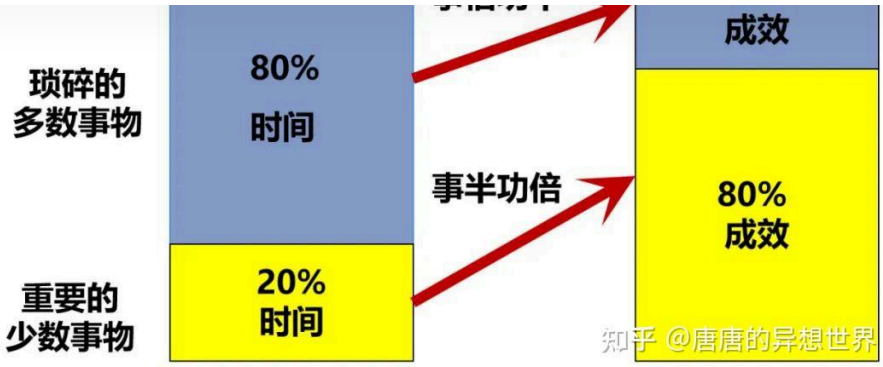
项目经理，既然是经理，那必然是具备经理人的能力，再辅以技术，如果不懂技术，最应该加强的能力是什么呢?

如果只能选择一个能力去加强，那就是**资源调配能力**。这个能力可以弥补你在技术上的不足。

咱们还是可以曲线救国的：

（1）了解产品和业务

著名的二八定律你一定都听过，用在这也同样适用。



找到问题的本质，解决本质问题，其他问题自然迎刃而解，对于公司来说，技术力量是最重要的吗？我相信在大多数公司都不是。

你可以不懂技术，但一定要摸透产品和业务。

想要在技术上赶超已绝无可能，还好对于企业来说，业务和产品才是根本，只要把你80%的时间花在这上面，成为团队里最懂业务的人，还有谁敢不听你的？

为什么要懂业务还有一个理由，项目经理的职责是做正确的事，基于业务和产品角度出发，就不会让团队偏离方向。

（2）了解团队能力，有很好的资源调配能力

不懂技术没关系，会用好这些做技术的人，能够团队成员要到资源，帮助他们成功，这就是你的本事。

什么意思？比如说，不会做技术性的项目规划，那就让懂技术的人愿意帮你做；缺资源缺人力，那就想办法让领导支持你……

让一切资源为我所用，是项目经理的境界之一，做到这个程度，你的管理一定非常轻松，难道还会去担心自己不懂技术，团队不服你吗？

（3）保持学习的动力。

不懂技术，但保持学习热情，跟“度娘”打成一片，归根结底，还是需要懂技术的，你至少要懂技术的流程和需要付出的时间，做好更合理的安排，自身也保持积极学习的状态，可以在团队中建立更好的影响力。

因为和技术相比，更重要的是PM能力的全面和平衡，项目经理大多活跃在团队、利益干系人之间，这里面的道是“平衡”，不是简单的用技术人员的线性思维就能够解决的，这算是非技术出身的一个优势吧！

不要忽视自己的直觉和感性判断，项目管理更重要的还是人，不是冷冰冰的代码，人的感觉特别重要。

最怕那种“虽然我不会，但是我只看结果，过程你们自己把握“的不求进步的PM，对团队来说是一场灾难，对于自己本身，也不会有任何提升。

三、最后从金融业业务和系统复杂度的角度来看，如何实现项目的成功

金融系统是架构界的珠穆朗玛峰，登峰不易。希望通过攀登上这座高峰，发现更好的风景。

还记得很多年前大学软件工程课上，老师介绍过几个最复杂的软件系统，有军用软件、操作系统，还有金融行业的软件。

军用软件的复杂度在于需要实时处理武器信号，操作系统的复杂度在于需要在功能的多样性和效率之间做一个良好的平衡，而金融软件的复杂度在于如何在软件系统的演进过程中保持并证明系统的

知乎

在银行业有一个说法是，如果有银行的核心系统宕机超过半个小时，肯定要被监管叫去喝茶了。这也充分说明了银行系统的复杂性，主要受业务特性影响和监管要求，银行系统复杂性的三个特点就是高可用、强一致性、强安全性。

1、高可用性

先来看看监管指引对于高可用性的具体要求

（1）商业银行数据中心监管指引

1. 商业银行应于取得金融许可证后两年内，设立生产中心；生产中心设立后两年内，设立灾备中心。

2. 商业银行数据中心应配置满足业务运营与管理要求的场地、基础设施、网络、信息系统和人员，并具备支持业务不间断服务的能力。

3. 总资产规模一千亿元人民币以上且跨省设立分支机构的法人商业银行，及省级农村信用联合社应设立异地模式灾备中心，重要信息系统灾难恢复能力应达到《信息安全技术信息系统灾难恢复规范》中定义的灾难恢复等级第5级（含）以上；其他法人商业银行应设立同城模式灾备中心并实现数据异地备份，重要信息系统灾难恢复能力应达到《信息安全技术信息系统灾难恢复规范》中定义的灾难恢复等级第4级（含）以上

（2）商业银行业务连续性监管指引

1. 商业银行应当综合分析重要业务运营中断可能产生的损失与业务恢复成本，结合业务服务时效性、服务周期等运行特点，确定重要业务恢复时间目标(业务RTO)、业务恢复点目标(业务RPO)，原则上，重要业务恢复时间目标不得大于4小时，重要业务恢复点目标不得大于半小时

2. 商业银行应当通过分析业务与信息系统的对应关系、信息系统之间的依赖关系，根据业务恢复时间目标、业务恢复点目标、业务应急响应时间、业务恢复的验证时间，确定信息系统恢复时间目标(信息系统RTO)、信息系统恢复点目标(信息系统RPO)，明确信息系统重要程度和恢复优先级别，并识别信息系统恢复所需的必要资源。

接下来看看商业银行为了满足高可用，具体的架构设计

正确的定义

金融行业覆盖的面非常广，不同子行业对容灾的要求会不一样。一种分类方式是按照用户专业性来分类，这种分法会将用户分为一般性用户和专业用户两大类。一般性用户指的是非从事专业金融类工作的用户。专业用户指的是以金融类工作谋生的用户。注意这里的用户并不限定为个人用户，其中还包括企业等机构用户。

通俗来讲一般性用户指的是 C 端用户，专业用户是 B 端用户。

对于一般用户来说，日常使用最多的金融服务是手机支付。如果手机支付系统出现了问题，支付公司进行了容灾处理之后，我们能接受的“正确”的容灾结果一般有两个。第一个是尽量在规定时间内恢复服务，比如在 10 秒内恢复。第二个是如果有限时间内不能恢复服务，那么要避免出现金额对不上的问题。比如说，我这边已经显示扣款了，商家那边却迟迟收不到钱的情况，就一定要避免。

专业用户也有一般用户的需求，但是相比一般用户又多了一项。专业用户在和金融机构对接时会签署服务合同。在合同内会规定金融机构在无法履约时的赔偿条例。比如说，对冲基金想通过证券公司的渠道抛售大量股票。如果此时证券公司系统崩溃，就会导致股票无法卖出。这个情况下，市场动荡带来的一部分经济损失可能需要由证券公司承担。

所以，我们综合两类用户需求的共性，可以这样来理解。从用户的角度来讲，正确容灾的第一个要求是服务质量协议（SLA，Service Level Agreement）的要求，规定一定时限内需要恢复服务。另一个要求是事务正确性的要求，特别是对事务 ACID 四个属性中 A 的要求，即 Atomicity，原子性。因为金融系统一旦出现金额问题后要给客户赔偿。那这两个要求之间是什么关系呢？如果金

临的是不可控的风险。这两个正确容灾要求的解决是需要成本的。如果实现容灾的成本大于赔偿金额，那么可以用日常营业利润的一部分作为容灾赔付基金。所以这里的正确并不是绝对的正确，而是相对的正确。你需要在服务质量、事务正确性和成本这三者之间做权衡。

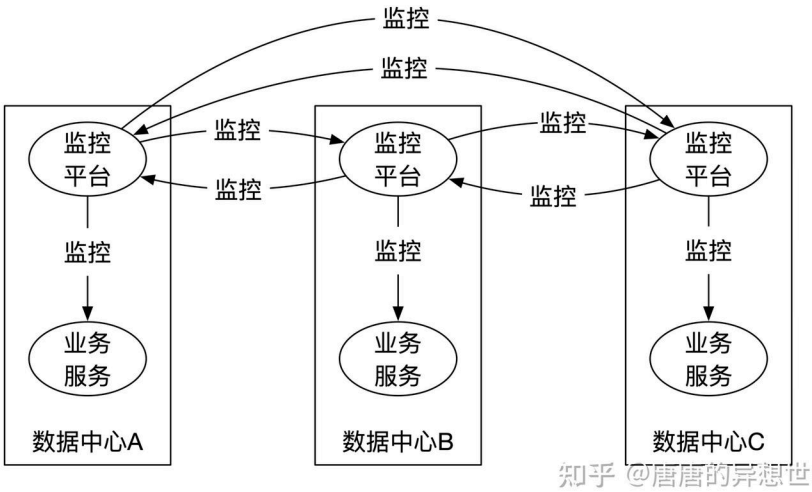
发现问题

人工监控

在解决容灾问题前，我们先要发现系统已经出了问题。虽然在互联网领域，现在的监控已经走向了自动化、甚至智能化，金融系统依然存在人工监控的情况。一个原因是一些特定金融业务的 SLA 要求很低，比如几天或者几个月，这时实现系统容灾成本大于收益。比如我们去实现资产证券化的业务，可能整个计算过程就在业务人员办公电脑的 Excel 里。如果电脑出问题了，重启一下就行，如果重启不了就换一位同事的电脑，而且这个过程可能一个月才做一次。一般来说，适合人工监控的一般是业务量小、客户少、金额高的金融业务。

系统监控让我们回到一般情况下的跨机房容灾监控问题。因为监控系统本身是一个很大的命题，在很多地方都有详细的介绍，所以在这里不做过多阐述。我重点给你讲讲，在解决跨机房容灾监控问题的时候，监控系统方面你需要注意的地方，主要有三点。

第一点是监控系统本身需要有跨机房容灾的能力。一种方式是用业务系统的跨机房容灾的架构来实现监控系统。虽然也可以解决问题，但是成本较高。这是因为监控数据是可以部分丢失的。我们在第 9 节课提到过数据具有时效性。监控数据和市场数据一样，时间越久，数据的价值越低。所以如果丢失了几秒钟的监控数据，你只要稍微等一下，刷新一下页面可能就会拿到最新数据。这样看来，我们就不需要为了数据中心级灾难这种小概率事件，采用高成本的容灾方案。因此，一个性价比较高的方案是监控数据本地异步备份，不同数据中心的监控平台彼此之间互相监控。

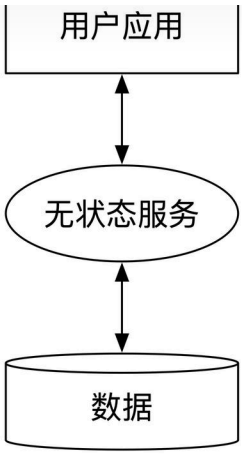


第二点是监控的延时。监控系统有推和拉两种获取监控数据的方式。推指的是业务系统主动将监控指标数据推送给监控系统，比如常见的心跳数据（heartbeat）。这么做实时性很强，但是对监控系统的峰值计算和存储能力要求很高。因此一般采用另一种折中的方式，让监控系统定期从业务系统里拉取监控指标。这样对硬件资源的要求基本固定，但是监控延时会稍微长一些，此时你需要和业务方仔细沟通金融业务需要的 SLA。

第三点是监控会伪报。当服务变多、网络变复杂之后，所有可能出现的异常都会出现。比如在防火墙配置出错的情况下，可能明明监控系统已经发现某服务不可用，但是该服务却在依然正确运行，想退出该服务的时候，退出指令也无法发送成功。因此，容灾处理需要考虑到在伪报情况下的正确性

容灾过程

在基于服务的架构下（SOA，Service Oriented Architecture），系统一般分为 3 层：用户应用层，无状态服务层和数据层。



因此在出现数据中心级别灾难的情况下，我们需要处理服务的容灾和数据的容灾。接下来，我们先看看服务容灾。服务容灾无状态服务容灾无状态服务的容灾相对简单，基本上什么都不做就可以了。一般来说，服务调度算法会将服务请求随机调度到不同的数据中心，因此当一个数据中心出现问题的情况下，用户服务在多次尝试之后，就会被调度到没有出现问题的数据中心。更进一步的优化方法是节省掉重复尝试的时间。如果数据中心内的服务均不可用，可以更新服务路由信息。这样新的请求不会被发送至出问题的数据中心节点，等到该数据中心恢复之后再加回到路由信息内。

消息重发问题

在容灾的过程中，无状态服务需要处理消息重发问题。如果表面上消失掉的无状态服务，其实依然还在处理服务，那我们该怎么办呢？一个常见的情况是服务成功处理了请求，但是由于防火墙问题无法将成功状态返回给用户。我们在第 8 节课一起研究过，怎么解决请求重发的问题。解决重发问题要求我们实现业务操作的幂等性（Idempotency），比如利用消息内的唯一标识符，或者使用 K/V 这种具有天然幂等性能力的数据结构

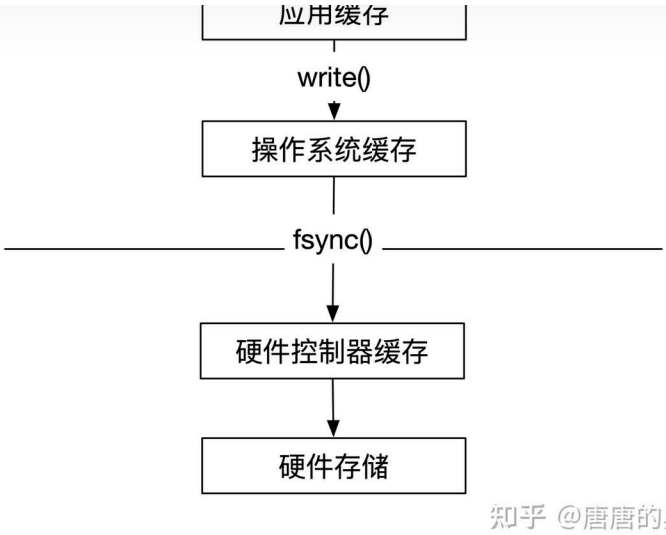
有状态服务容灾

金融行业的低延时场景通常会采用有状态的服务，比如外汇交易所或者股票交易所。这时候可以将有状态服务节点当作数据节点来处理

数据容灾

数据容灾一般需要数据系统自己来处理，容灾作为数据系统的一种自有的能力。这里我们重点看看金融系统常见的两个方案，分别是两地三中心和三地五中心的实时容灾方案。

单节点容灾为了满足金融级的容灾要求，我们需要先保证单机节点具有一定的容灾能力。单机节点容灾指的是，数据节点重启后能恢复之前所有数据。在这里有一些细节需要你注意。通常在写入数据文件时，数据并没有被写入到硬盘中，此时数据会保存在操作系统内的缓存。当这个缓存满了，或者等待了一定时间之后（比如 30 秒钟），操作系统会将这个缓存写入到硬件控制器的缓存。用户也可以通过 fsync 的系统调用来主动完成这个过程，但是需要小心的是此时只写入到了硬件控制器缓存，并没有写到了硬盘。如果这时候机器断电，依然有可能出现数据丢失。



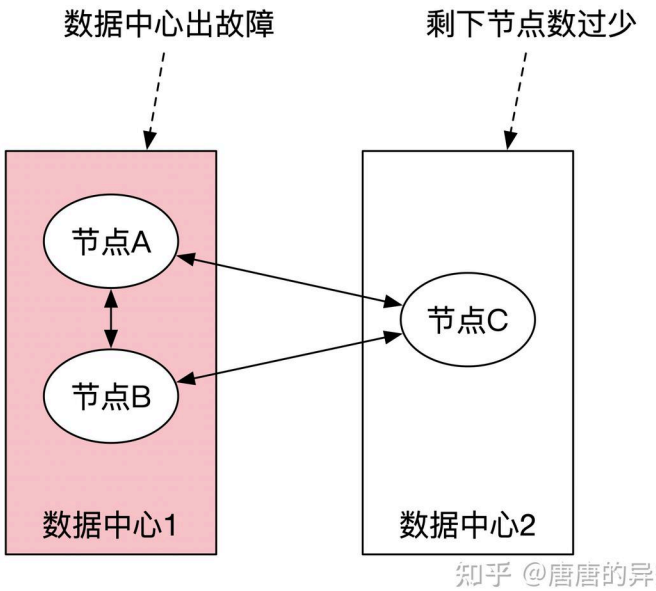
那么怎么处理这种小概率事件呢？也有两种方式。一种方式是用企业级硬件。

企业级硬盘可能会携带一小块电池。当机器断电之后，这块电池能保证硬件控制器内的数据都能写入到硬盘中，这是用钱来换正确性。另一种方式是不管不问。如果我们已经确定了，一定会用一组集群来实现多节点容灾，这种情况下，因为断电而产生的数据丢失可以看作是单节点故障来处理。这也是用钱来换正确性的思路，Kafka 就是用了这种假设，甚至 Kafka 连 fsync 都节省掉了。所以单节点容灾到什么程度取决于你有多少钱

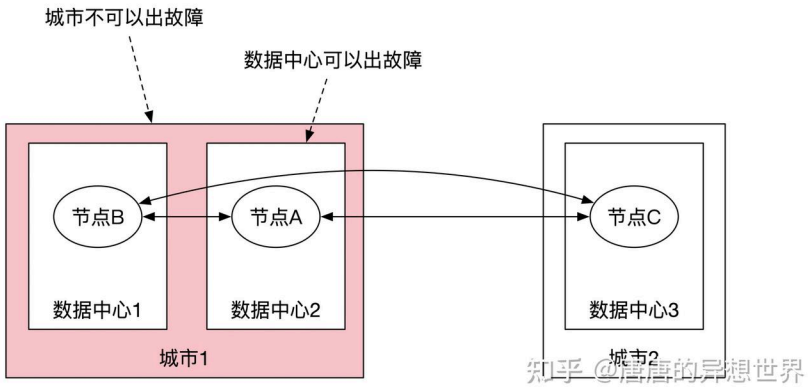
多节点跨机房容灾

跨机房部署两地三中心或者三地五中心的容灾利用了共识（consensus）算法的能力。我们在第 16 节课详细讲过 Raft 共识算法，如果你记不清了，可以做个回顾。Raft 的部署一般有 3 个备份或者 5 个备份两种选项。偶尔也会出现 9 个备份的选择，比如 Google。我们先来看看有 3 个备份的情况。这时候有两种选择。一种选择是一个数据中心有 2 个节点，另一个数据中心只有 1 个节点。这样只能保证单节点级别的容灾，不能保证数据中心级的容灾。

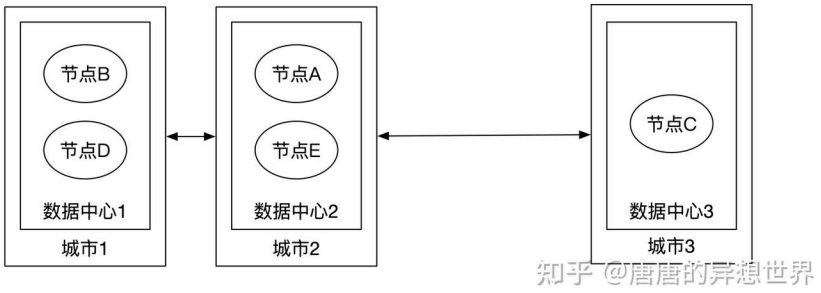
比如下面这幅图展示了一个典型的部署。Raft 算法要求有一大半的机器在线才能正常工作，因此 3 个备份的情况下至少需要 2 台节点在线。如果下图的数据中心 1 整个出了故障，那么只剩下一个节点在数据中心 2，算法无法正常工作。



灾，但是不能满足城市级容灾。比如下图所示。如果城市 1 因为道路施工挖断了光纤，城市 2 就不能保证 Raft 协议正常工作。



5 备份的情况和 3 备份的情况类似，但是选择更多。其中有一些选择能达到城市级别的容灾。我们举个例子来理解。如下图所示，这时候 5 个节点基本平分到了 3 个城市的数据中心。你可以算一算，这时候任何一个城市网络出现了问题，Raft 协议都能正常运行。这就是常见的三地五中心部署，也是普遍使用的较高容灾级别的部署方式。



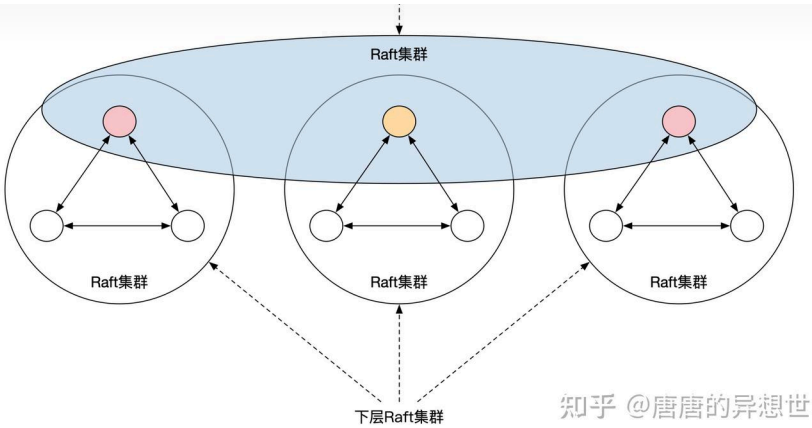
总结一下。在使用多个节点容灾的情况下，两地三中心只能达到数据中心级别的容灾。如果需要达到城市级别的容灾，需要三地五中心部署。

客户端处理

我们在进行容灾分析的时候，一般会侧重处理服务器端的各种特殊情况，这时候很容易忽略客户端。从原理上来说，如果服务器端容灾做得正确，就不会出现数据正确性的问题。但是我们在开头提到过，容灾除了正确性之外，还有一个服务质量协议的问题，我们还需要尽量减少无法提供服务的时间长度。因此，我还是要强调一下正确的客户端行为。当服务器出现问题的时候，Raft 协议自动换主之后，客户端一定要用正确的方式来找到新的主节点，这样会大大减少容灾的延时。

除了三地五中心之外，还有一种容灾能力更高的部署方式，那就是三地九中心，Google 曾经采用过这种部署方式。三地九中心并不是直接部署 9 个 Raft 节点，而是将 Raft 节点分为了两层。下面一层按照 3 个一组分为了 3 组，分别放在 3 个数据中心。每个数据中心的 3 个节点刚好组成一个 Raft 集群，通过 Raft 选主的方式选出来一个主节点。这样 3 个数据中心就一共有 3 个主节点。这 3 个主节点之间刚好也可以形成一个 Raft 集群，再选出一个级别更高的 Raft 主节点。这个唯一的主节点负责代表集群对外提供服务。下面这幅图展示了三地九中心的部署方式。

知乎



最后结合银行的高可用案例，看看银行业从网络层到应用层的高可用策略

编辑于 2023-04-18 20:18 · IP 属地上海

项目管理　项目管理工具

理性发言，友善互动

还没有评论，发表第一个评论吧

推荐阅读

浅谈需求管理在银行业IT项目中的重要性

维普时代

银行业科技项目管理发展策略分析

维普时代

信息系统项目管理师快速记忆口诀

看过信息系统项目管理师教材和真题的考友们都知道，考试涉及的内容非常烦杂，涵盖面很广泛，几乎涵盖了计算机和项目管理的大部分内容，除此之外，它还有一部分自身特有的内容，如管理运筹学...

科科过

IT项目

项目管理
供某项
的临时
的产品、
人）逐
需要不

拖懒之