

SACC

2021 中国系统架构师大会

SYSTEM ARCHITECT CONFERENCE CHINA 2021

数字转型 架构重塑

IT168.com

ChinaUnix

ITPUB

云上会议 网络直播 | 2021.5.20-2021.5.22

架构师所需的硬实力与软技能

演讲人：苏仕祥

目录

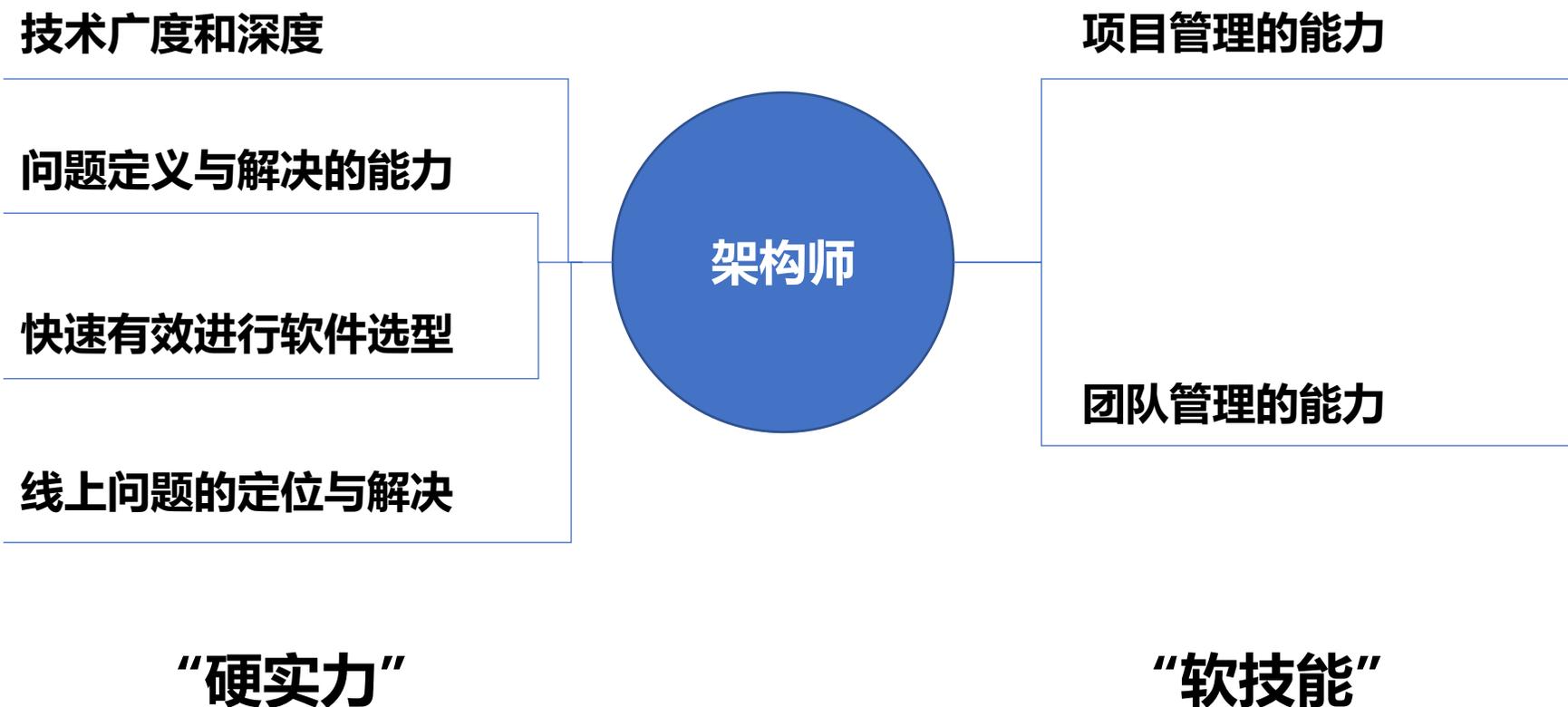
01 概述

02 架构师所需的硬实力

03 架构师所需的软技能

01 概述

01 概述



02 架构师所需的硬实力

02 架构师所需的硬实力—技术广度和深度



技术学广还是学精？

02 架构师所需的硬实力—技术广度和深度

CNCF Cloud Native Landscape
2019-09-10T05:15:36Z c43ba46

Overwhelmed? Please see the CNCF Trail Map. That and the interactive landscape are at l.cncf.io

Database | Streaming & Messaging | Application Definition & Image Build | Continuous Integration & Delivery | Platform | Observability and Analysis

App Definition and Development | Orchestration & Management | Runtime | Provisioning | Special | Kubernetes Certified Service Provider | Kubernetes Training Partner

Scheduling & Orchestration | Coordination & Service Discovery | Remote Procedure Call | Service Proxy | API Gateway | Service Mesh | Cloud Native Storage | Container Runtime | Cloud Native Network | Automation & Configuration | Container Registry | Security & Compliance | Key Management | PaaS/Container Service | Serverless | Members

Monitoring | Logging | Tracing | Chaos Engineering

Greyed logos are not open source

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

l.cncf.io

CLOUD NATIVE COMPUTING FOUNDATION
CLOUD NATIVE LANDSCAPE

Special

Kubernetes Certified Service Provider

Kubernetes Training Partner

Members

学广怎么学？

02 架构师所需的硬实力—技术广度和深度

学深怎么学？

Spring学深能学到哪些？

MySQL学深能学到哪些？

中间件（Mycat、RocketMQ、Redis等）学深能学到哪些？

02 架构师所需的硬实力—技术广度和深度



- 1 JAVA语言（数据结构、算法、类加载、多线程等）
- 2 网络处理（Http协议、IO模型）
- 3 应用部署运维（操作系统、负载均衡、高可用）

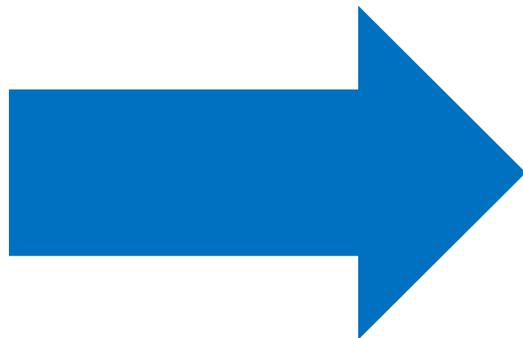
02 架构师所需的硬实力—技术广度和深度



- 1 C++语言（数据结构、算法等）
- 2 网络处理（MySQL协议、IO模型）
- 3 应用部署运维（高可用、高可靠、数据复制、操作系统）

02 架构师所需的硬实力—技术广度和深度

计算机技术



数据结构

算法

网络

操作系统

02 架构师所需的硬实力—技术广度和深度



02 架构师所需的硬实力—技术广度和深度



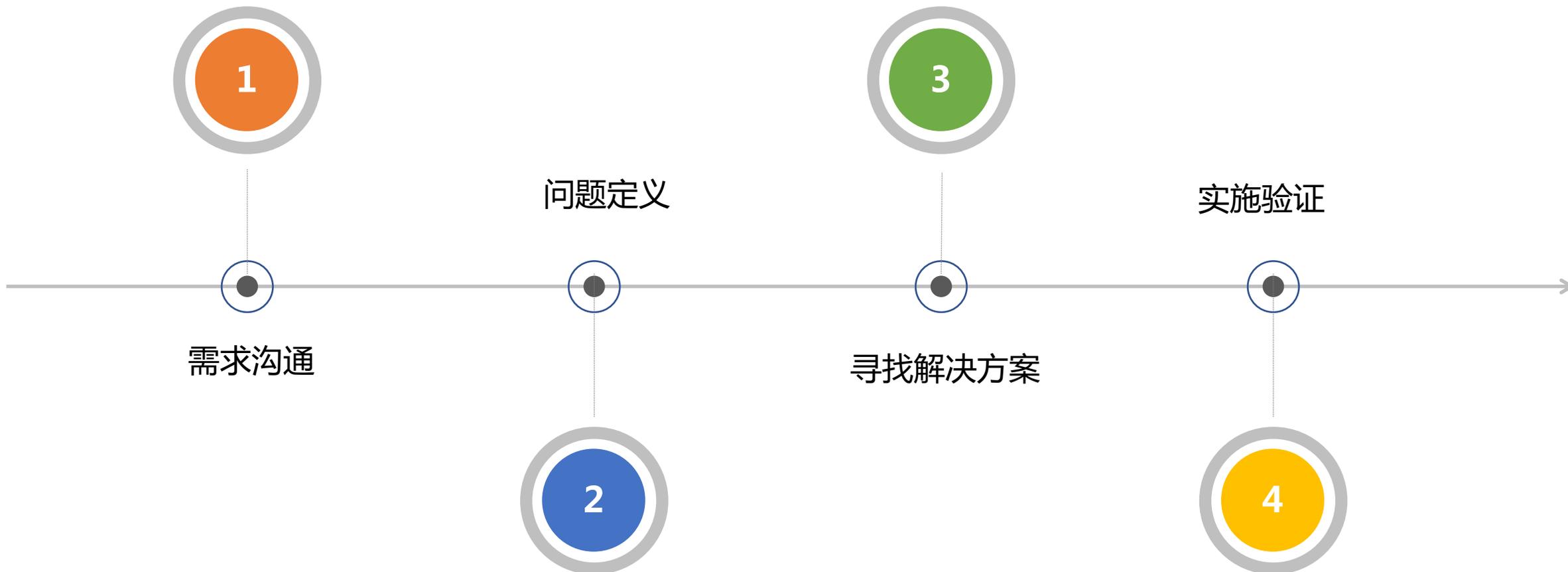
融会贯通，一通百通

02 架构师所需的硬实力—问题定义与解决的能力



在解决问题之前，我们首先需要知道要解决的问题是什么？

02 架构师所需的硬实力—问题定义与解决的能力



02 架构师所需的硬实力—问题定义与解决的能力

如何提高问题定义的能力？

- 沟通
 - 认真聆听
 - 自我表述
 - 总结确认
- 持续学习
 - 扩展知识面
 - 不自我设限
- 抽象概括

如何提高问题解决的能力？

- 信息收集
- 持续学习
- 团队能力
 - 集思广益

02 架构师所需的硬实力—快速有效进行软件选型

选型的需求来源于业务诉求

选型总是在业务驱动下进行的。

我相信业务用的好好的，没有性能瓶颈，没有合规要求，也没有新需求要开发，我们也不会想着换个框架或组件。

当业务有真正的诉求的时候，比如有数据同步，业务解藕，削峰填谷，分布式事务等需求时，我们就需要调研相应的解决方案了。

02 架构师所需的硬实力—快速有效进行软件选型

选择调研对象

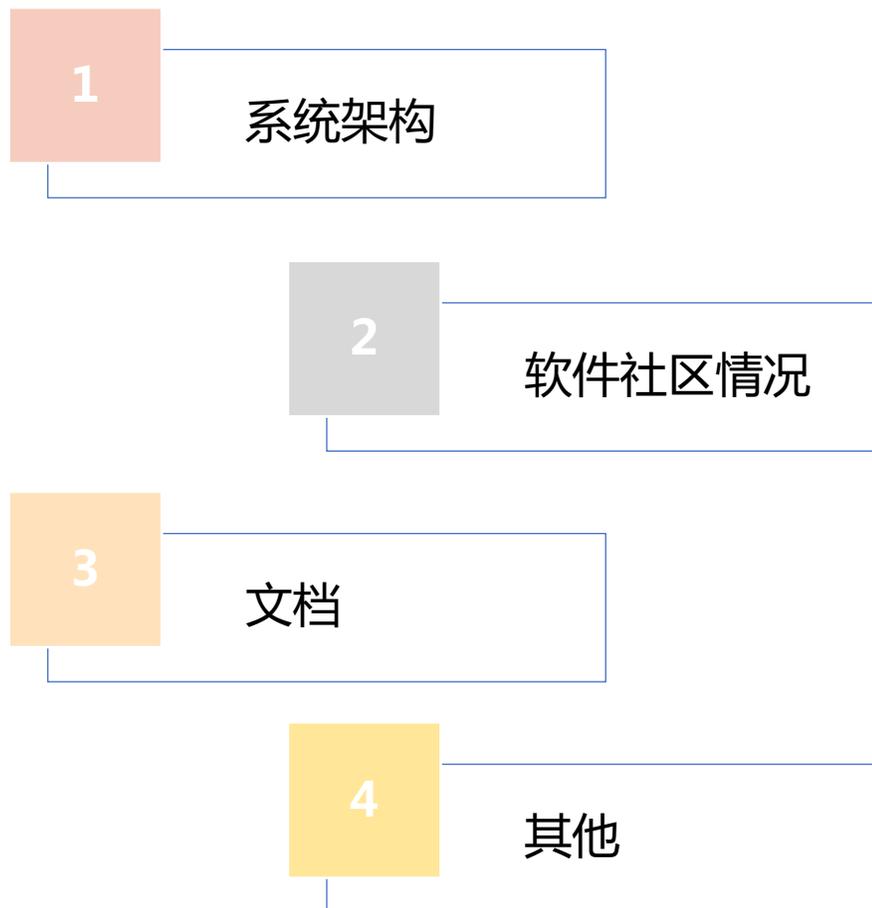
在开始选型前，我们需要知道有哪些待调研的对象。

想用消息中间件，需要调研的对象可能有RabbitMQ，RocketMQ，Kafka等。

有数据同步的需求，需要调研的对象可能有otter，Datax，canal等。
想用rpc服务，调研的对象可能有Dubbo、gRPC、Thrift等。

02 架构师所需的硬实力—快速有效进行软件选型

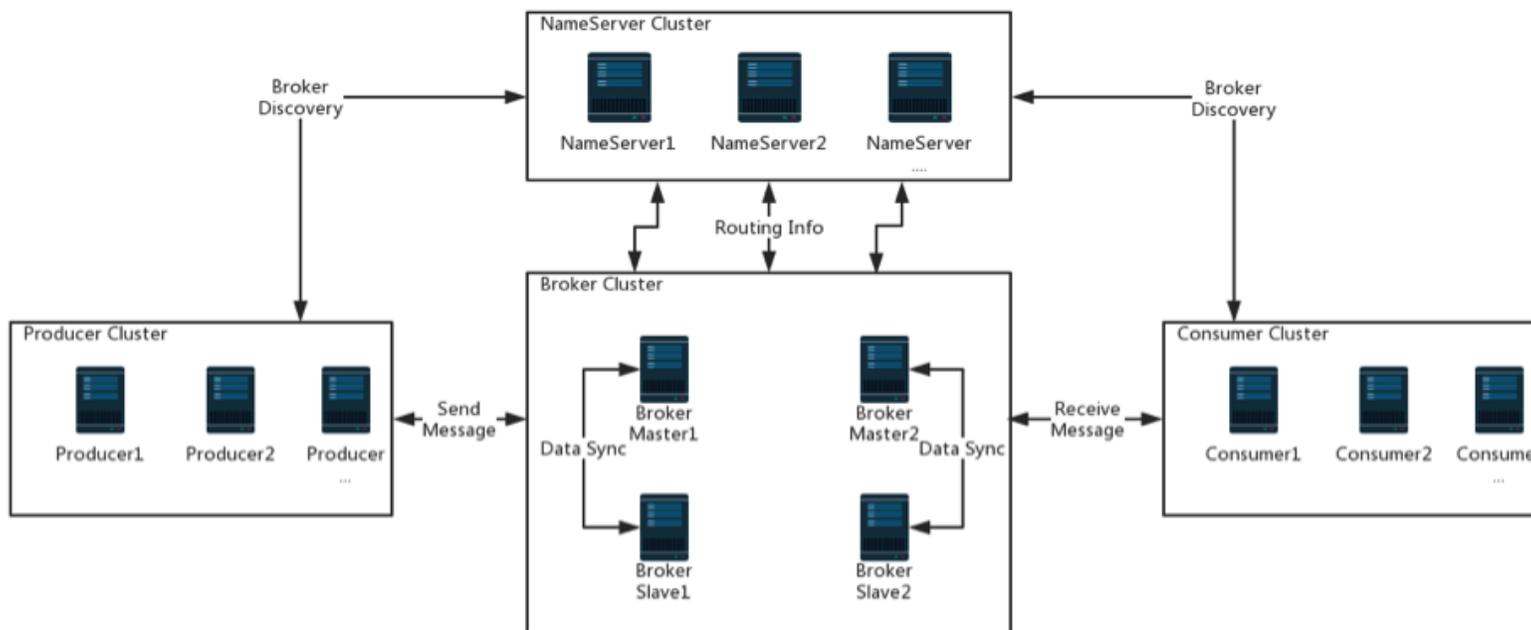
开始调研



02 架构师所需的硬实力—快速有效进行软件选型

开始调研-系统架构

软件的系统架构很大程度上决定了软件的简单或复杂、可扩展性、高可用性等较难改变的属性。



02 架构师所需的硬实力—快速有效进行软件选型

开始调研-软件的社区情况

活跃的社区对于开源软件来讲很重要，因为活跃的社区说明软件的用户很多，用户多就有很多的实践经验可以参考交流，在你遇到问题的时候能有个交流的平台。

社区情况一般看软件的最近一次提交是什么时候，issue的数量及回复情况，更进一步可以看软件的邮件列表内容。

02 架构师所需的硬实力—快速有效进行软件选型

开始调研-文档

产品做的再好，没有文档，也没人会用。

详细的产品文档包括但不限于软件架构，用户手册，性能测试，常见问答等

Spring Framework Documentation

Version 5.3.4

What's New, Upgrade Notes, Supported Versions, and other topics, independent of release cadence, are maintained externally on the project's [Github Wiki](#).

- Overview** history, design philosophy, feedback, getting started.
- Core** IoC Container, Events, Resources, i18n, Validation, Data Binding, Type Conversion, SpEL, AOP.
- Testing** Mock Objects, TestContext Framework, Spring MVC Test, WebTestClient.
- Data Access** Transactions, DAO Support, JDBC, R2DBC, O/R Mapping, XML Marshalling.
- Web Servlet** Spring MVC, WebSocket, SockJS, STOMP Messaging.
- Web Reactive** Spring WebFlux, WebClient, WebSocket, RSocket.
- Integration** Remoting, JMS, JCA, JMX, Email, Tasks, Scheduling, Caching.
- Languages** Kotlin, Groovy, Dynamic Languages.

02 架构师所需的硬实力—快速有效进行软件选型

开始调研-其他

包括但不限于合规性，法律，生态工具等内容，比如如果使用的开源软件协议为 GPL，按照协议规定，所有的二次开发都应该再次开源出来。

02 架构师所需的硬实力—快速有效进行软件选型

特性或功能验证

调研完产品后，我们对软件应该有了个大概了解了，此刻我们需要回到业务上来。

产品的特性或功能能否满足业务的需求，这才是决定我们选型的最重要原因。

一般的原则是，能够满足当前以及可预见的未来的需求的软件，是最合适的，因为这样的软件又能满足需求，又不至于太过复杂。

此阶段一般会进入到体力活环节，需要各种测试，比如基本功能测试，高可用测试，性能测试等。在测试的过程中也尽可能的将测试脚本化、自动化，因为这个过程可能会重复多次，没人想一次次的手动来做。

应用阶段

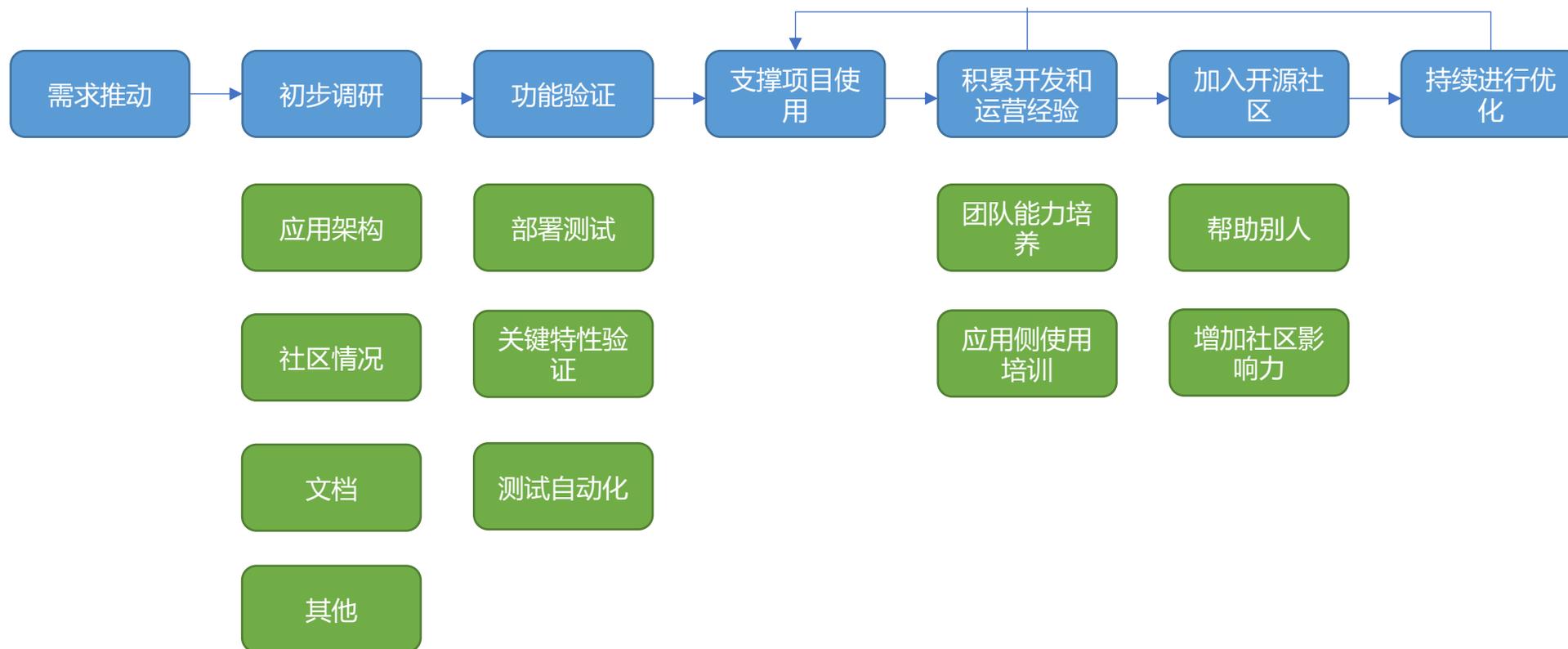
如果软件是基础软件比如数据库之类，一般会先进行试点应用，因为基础软件的影响面比较大，而选择试点应用，既可以规避大的风险，也能够通过实践积累起经验，方便后续的全面升级。

如果软件只是当前业务使用，影响较小，那么要有自信直接上，不要怂~

帮助别人

经历了软件调研，功能特性验证，到项目的正式使用，可以说现在你也成为了软件社区中的一员，那么不要忘了分享你的使用经验，帮助其他软件使用者，毕竟一开始也是社区帮你解决了问题。

02 架构师所需的硬实力—快速有效进行软件选型



过程资产文档化

02 架构师所需的硬实力—线上问题的定位与处理



架构师需要有救火的能力。

02 架构师所需的硬实力—线上问题的定位与处理

如何提升救火的能力？



正确归因



复现-修复-测试-回归

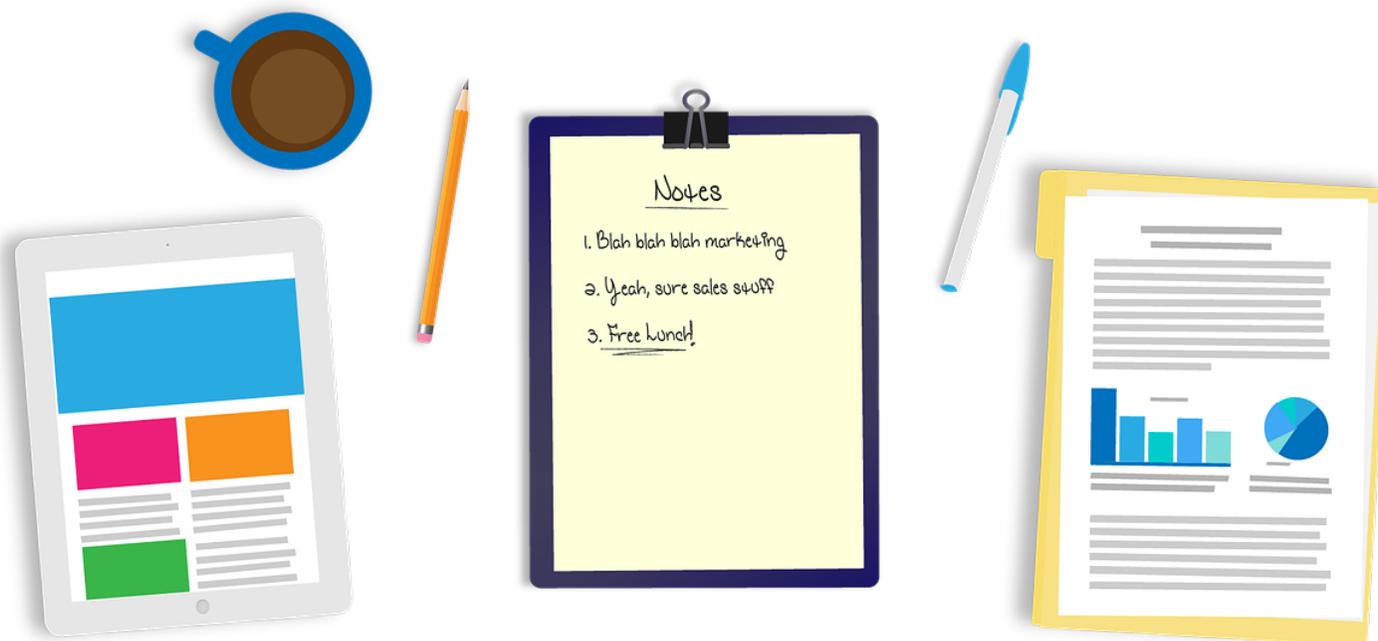


持续学习知识，实践中积累知识

02 架构师所需的软技能

02 架构师所需的软技能—项目管理的能力

我们是如何开始一个项目的？



02 架构师所需的软技能—项目管理的能力

项目管理的各个领域



02 架构师所需的软技能—项目管理的能力

项目范围管理

项目范围管理最重要的是确认项目产出物。

02 架构师所需的软技能—项目管理的能力

项目时间管理-工作量评估

项目时间管理的核心在正确评估工作量。

我们来看一下项目工作量评估的几种情况。

第一种情况，能准确的评估出工作量。这种情况下，相应的项目对于公司来讲可以说是轻车熟路了，有足够的项目信息可供参考。

第二种情况，能大概评估出工作量。此种情况，多发生在公司拓展业务时。公司以前是做智慧交通的，现在接到了智慧医疗的项目，组织内没有足够的信息可供参考。

第三种情况，无法评估出工作量。原始创新型项目，没有任何信息可供参考。

02 架构师所需的软技能—项目管理的能力

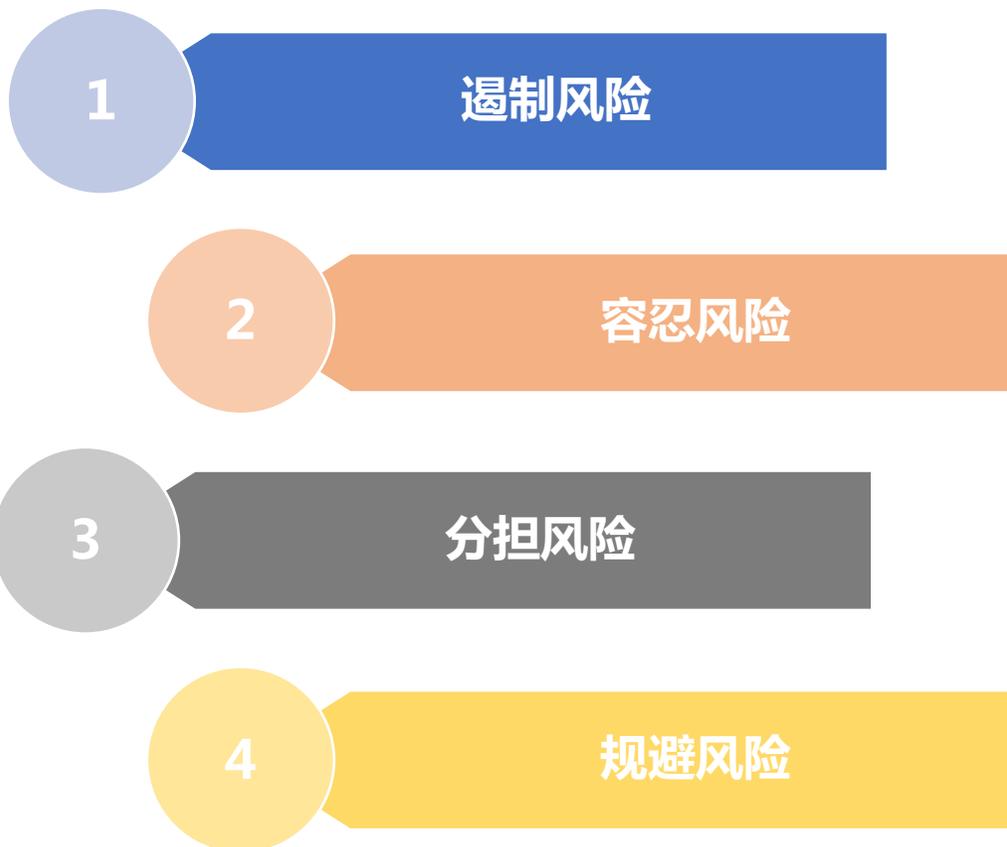
项目时间管理-如何正确评估工作量

充分收集信息

02 架构师所需的软技能—项目管理的能力

项目风险管理

项目风险如何应对？



02 架构师所需的软技能—项目管理的能力

项目沟通管理

项目沟通主要是在汇报项目进度上。

项目沟通原则——简单直观。能用图的不要用表，能用表的不要用文字。

内容	8月									
	1日-3日	4日-6日	7日-9日	10日-12日	13日-15日	16日-18日	19日-21日	22日-24日	25日-27日	
项目启动 (1日-9日)	黄色									
项目规划 (10日-27日)				绿色						
项目执行 (13日-31日)					红色					
项目收尾 (13日-31日)						红色				

02 架构师所需的软技能—团队管理的能力



闲适总是对人有好处的。

02 架构师所需的软技能—团队管理的能力

我们为什么要努力？

1

生活所迫

2

远大目标

3

兴趣爱好

02 架构师所需的软技能—团队管理的能力

什么是成功的团队管理？

满足各利益方期望。



实践、成长、
收获的期望



完成经营目标、获取
盈利的期望



成长、收获的期望

02 架构师所需的软技能——团队管理的能力

什么是成功的团队管理？

满足各利益方期望。

个人、公司、团队管理者，利益并不冲突，完全能够实现共赢。

02 架构师所需的软技能—团队管理的能力

如何实现共赢？



谢谢！



聊技术，不止于技术。

个人公众号：WU双