

构建微服务云原生应用

架构师杨波



扫码试看/订阅

《Spring Boot & Kubernetes 云原生微服务实践》

CHAPTER

02

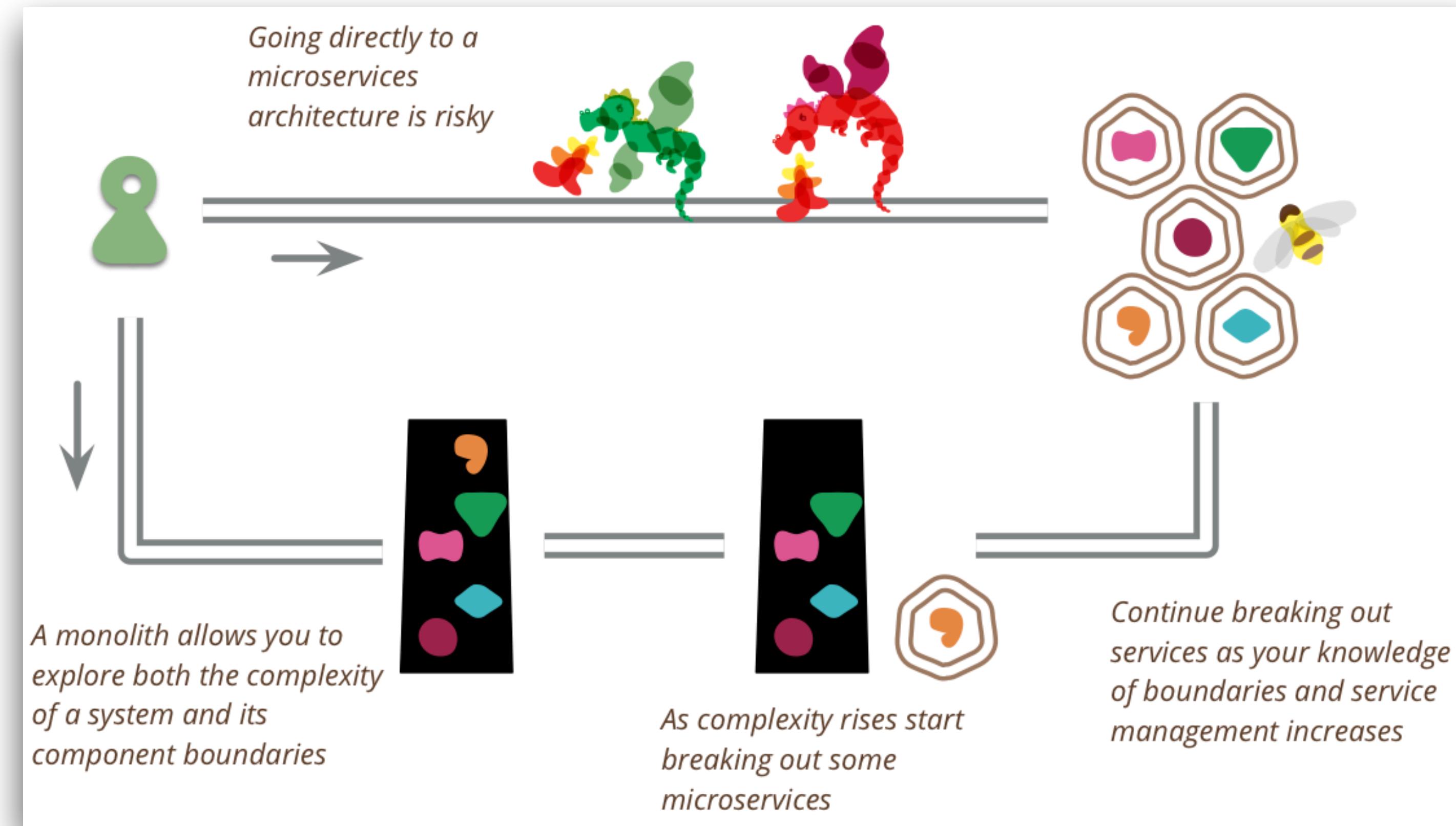
系统架构设计和技术栈选型



第 1 部分

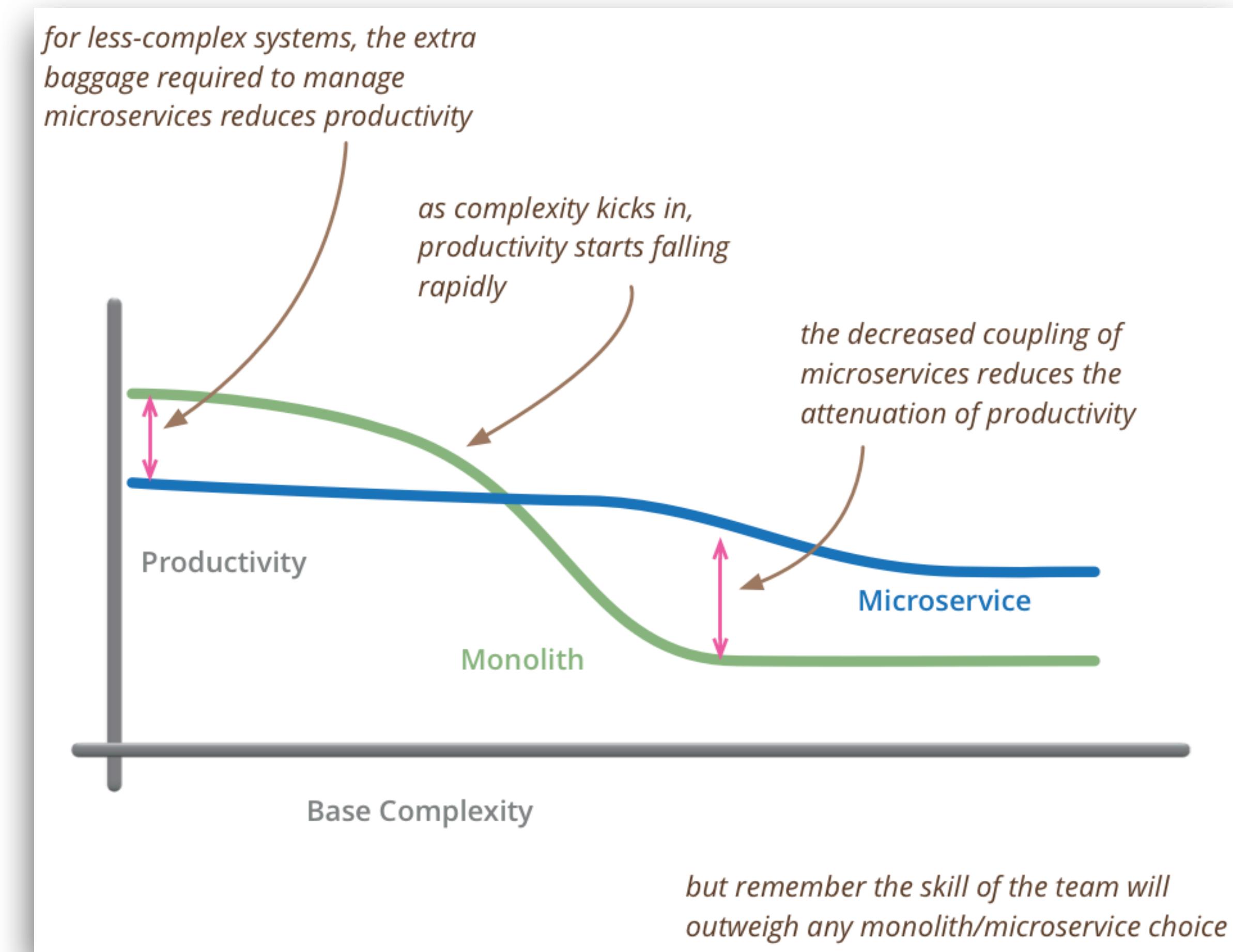
为何采用微服务架构

观点1:单块优先



<https://martinfowler.com/bliki/MonolithFirst.html>

微服务引入时机



<https://martinfowler.com/bliki/MicroservicePremium.html>

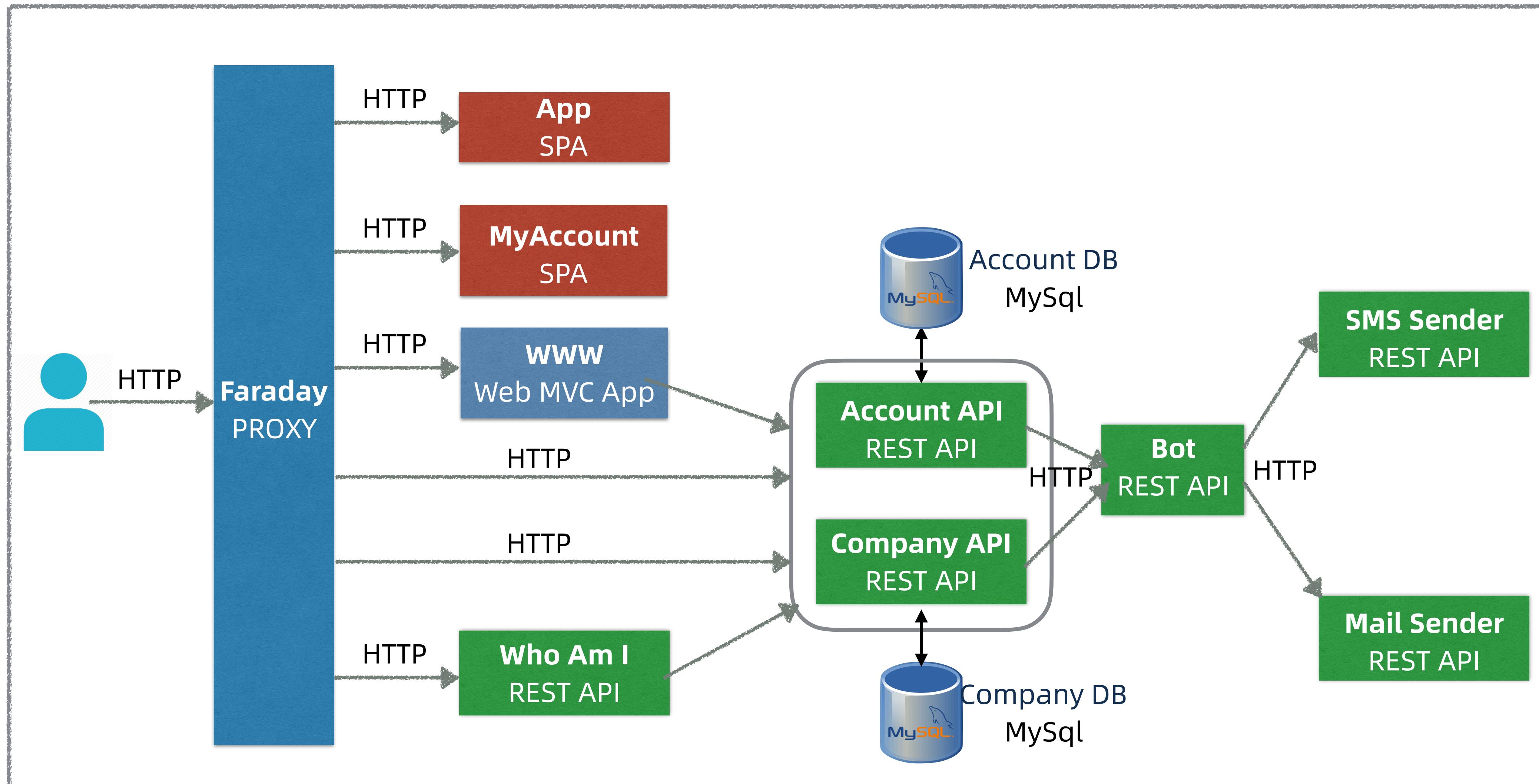
观点2：微服务优先



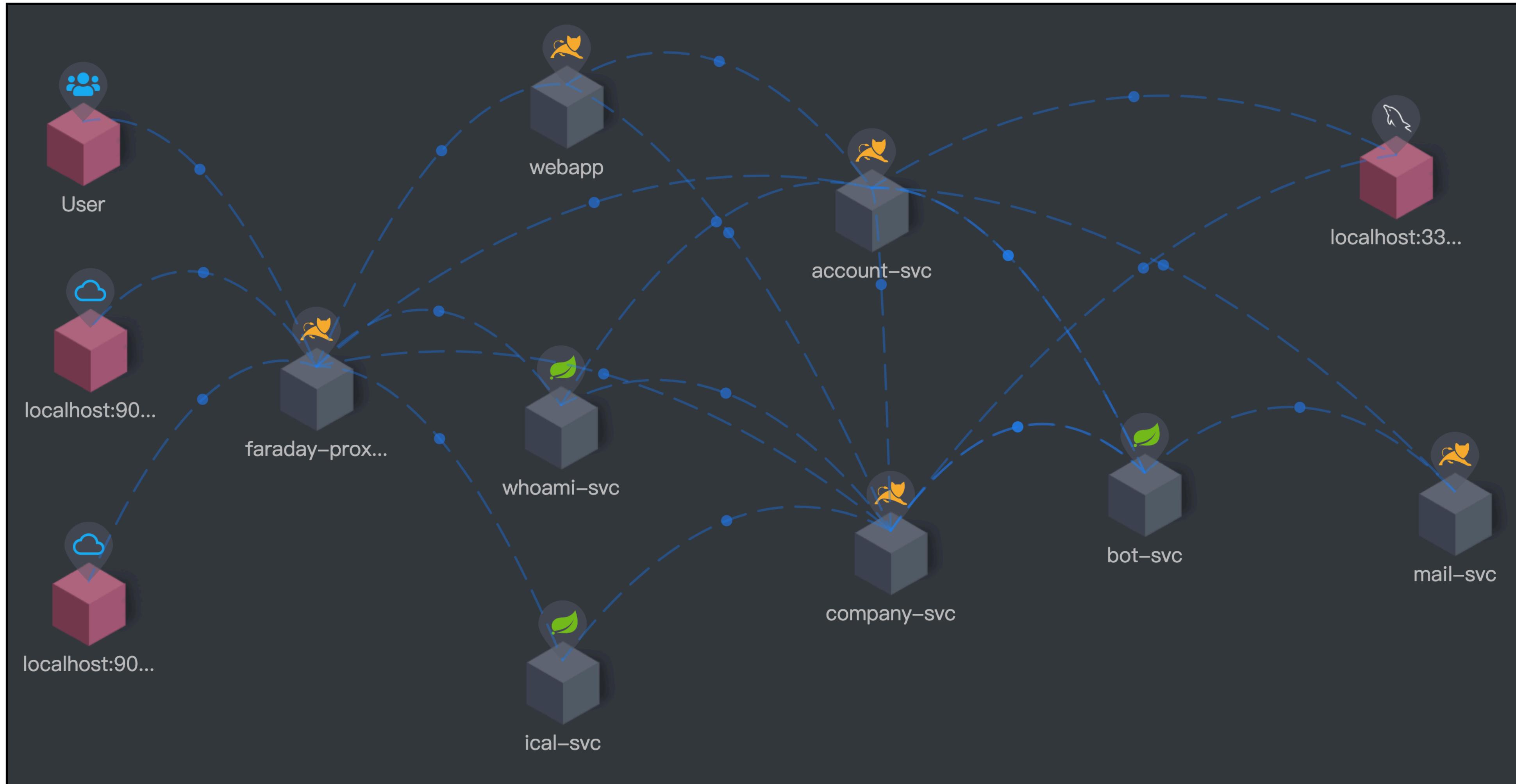
第 2 部分

架构设计和技术栈选型

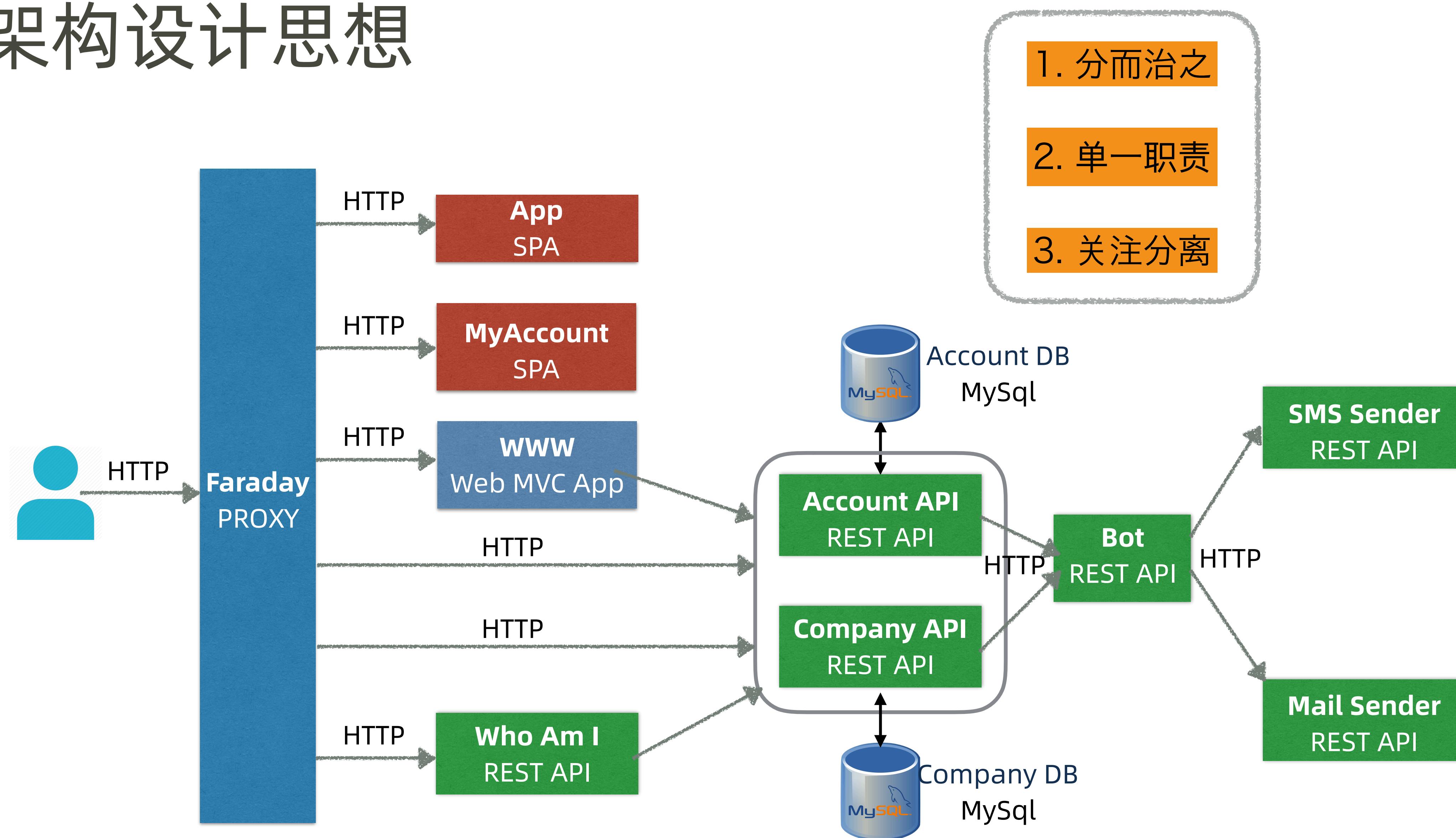
总体架构设计



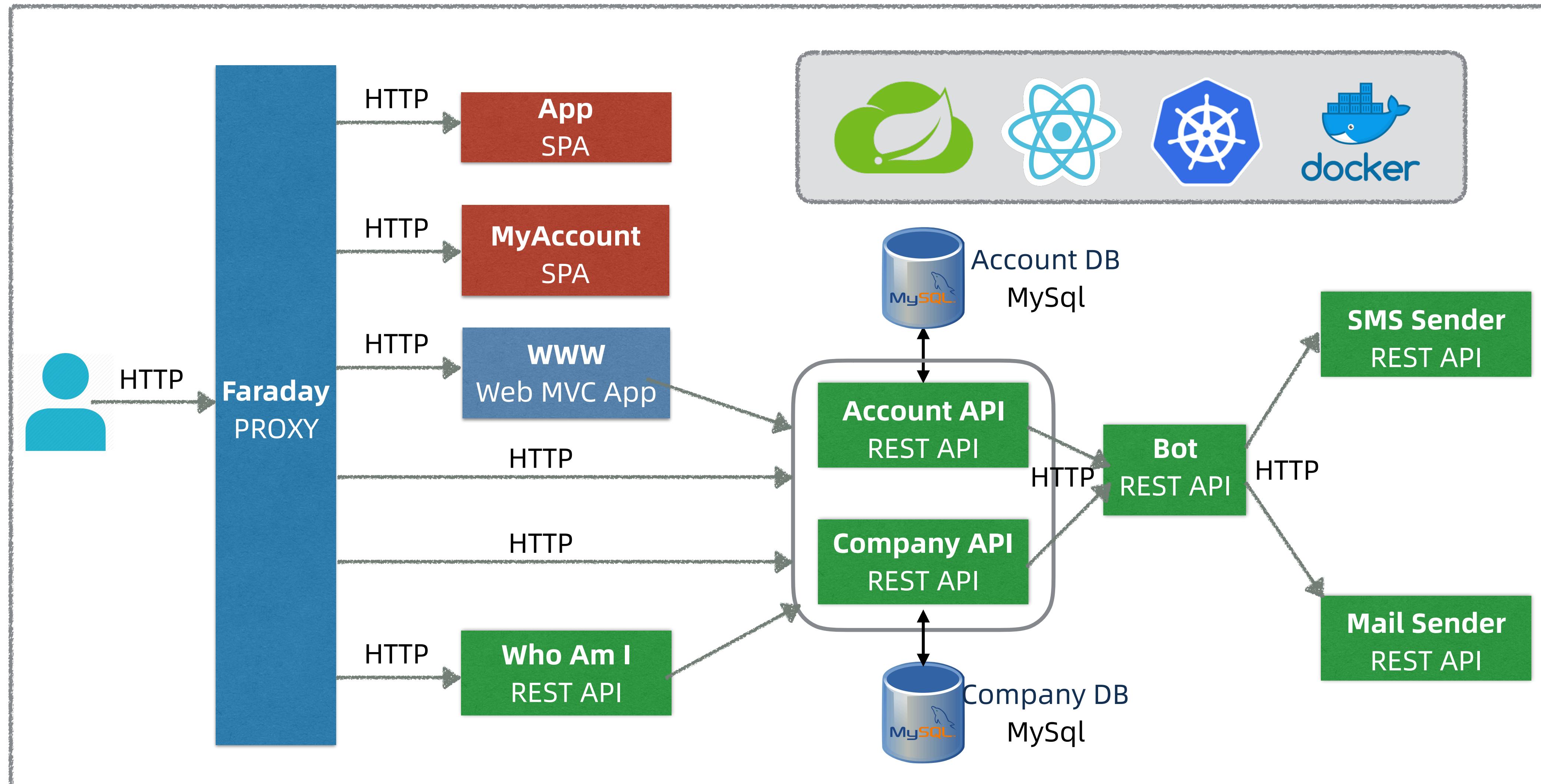
Skywalking 依赖监控图



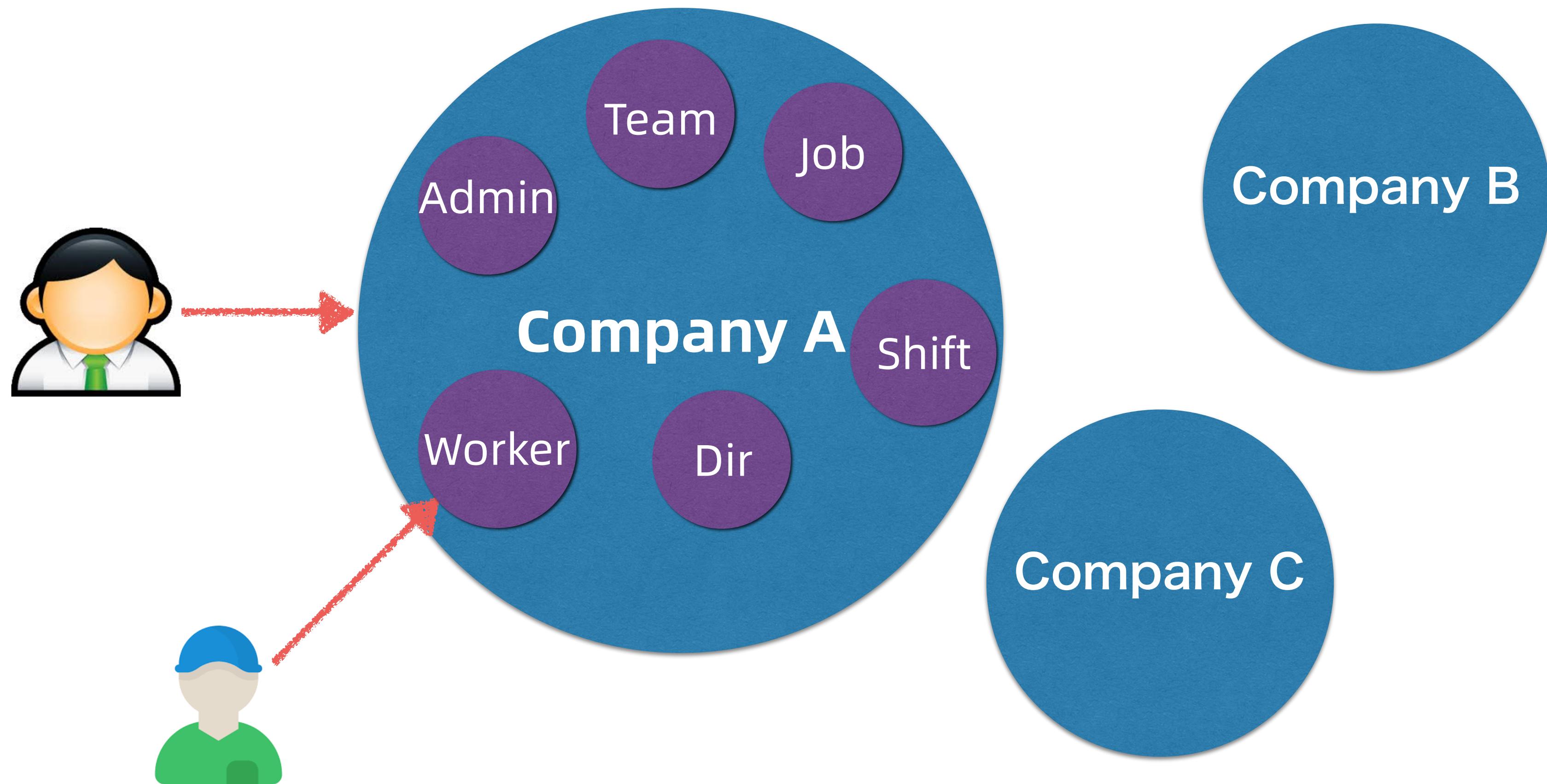
架构设计思想



技术栈选型



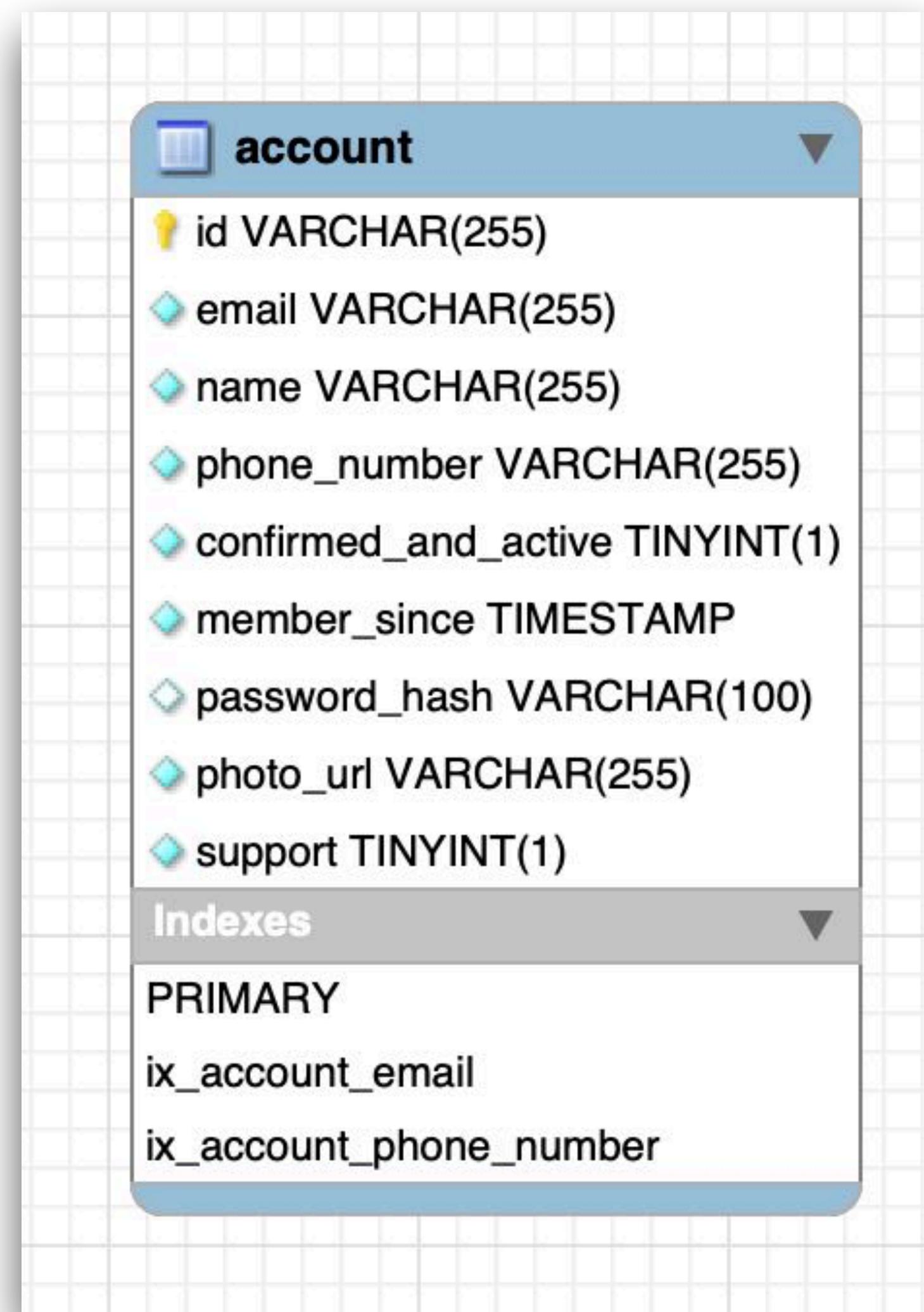
SaaS 多租户设计



第 3 部分

数据和接口模型设计 ~ 账户服务

账户数据模型



账户接口

操作	HTTP方法	功能
createAccount	POST	创建新账户
getOrCreateAccount	POST	获取或创建(如不存在)账户
get	GET	通过用户id获取已有账户
getAccountByPhonenumber	GET	通过电话号码获取已有账户
list	GET	获取现有账户列表(内部使用)
updatePassword	PUT	更新密码
verifyPassword	POST	校验密码
requestPasswordReset	POST	请求密码重置
update	PUT	更新账户
syncUser	POST	同步信息到Intercom客户系统
trackEvent	POST	同步用户事件到Intercom客服系统

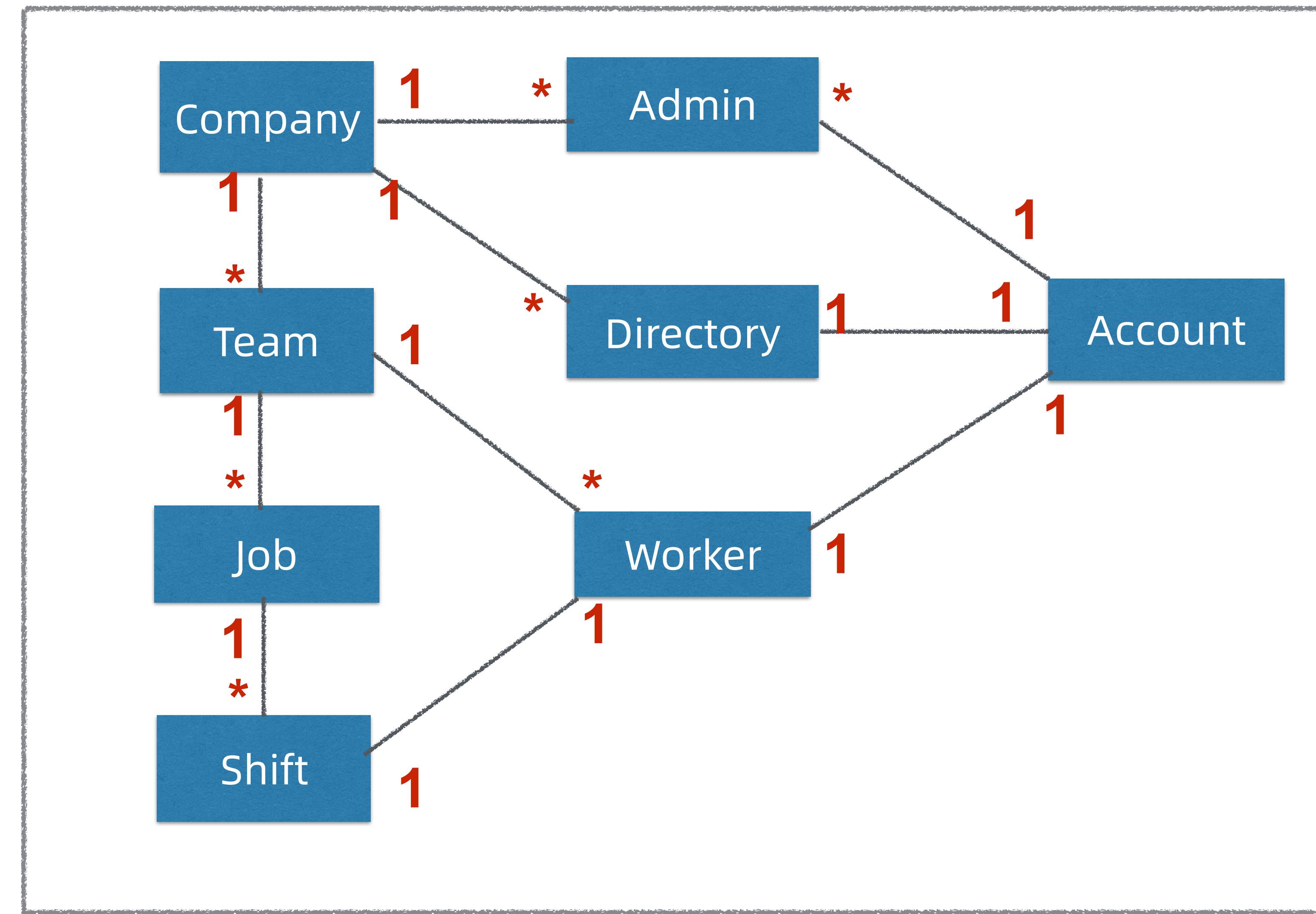
第 4 部分

数据和接口模型设计 ~ 公司服务

公司数据模型



实体关系ER图(简化)



公司Company服务接口模型

操作	HTTP方法	功能
createCompany	POST	创建公司
listCompanies	GET	获取现有公司列表(内部使用)
getCompany	GET	通过id获取公司
updateCompany	PUT	更新公司信息

公司管理员Admin服务接口模型

操作	HTTP方法	功能
createAdmin	POST	创建用户和公司间的管理员关系
listAdmins	GET	通过公司id获取管理员用户列表
getAdmin	GET	通过公司id和用户id获取管理员关系
getAdminOf	GET	通过用户id获取其管理的公司列表
deleteAdmin	DELETE	删除用户和公司间的管理员关系

员工目录Directory服务接口模型

操作	HTTP方法	功能
createDirectory	POST	将某用户添加到公司员工目录中
listDirectories	GET	列出某公司id下的所有员工目录项
getDirectoryEntry	GET	通过公司id和用户id查询某员工目录项
getAssociations	GET	获取某公司id下的所有员工目录项，包括是否管理员，对应团队信息
updateDirectoryEntry	PUT	更新员工目录项

团队Team服务接口模型

操作	HTTP方法	功能
createTeam	POST	创建团队
listTeams	GET	列出某公司id下的所有团队
getTeam	GET	通过公司id和团队id获取团队信息
updateTeam	PUT	更新团队信息
getWorkerTeamInfo	GET	通过公司id和用户id查询该员工隶属团队

雇员Worker服务接口模型

操作	HTTP方法	功能
createWorker	POST	建立某用户和某公司/团队间的雇员关系
listWorkers	GET	列出某公司id和团队id下的所有雇员目录项
getWorker	GET	获取某公司id、团队id和用户id的雇员目录项
getWorkerOf	GET	获取某用户id所隶属的团队
deleteWorker	DELETE	删除某用户和某公司/团队间的雇员关系

任务Job服务接口模型

操作	HTTP方法	功能
createJob	POST	为某公司/团队新建任务
listJobs	GET	列出某公司/团队下的所有任务
getJob	GET	获取某公司id, 团队id和任务id所对应的任务
updateJob	PUT	更新任务信息

班次Shift服务接口模型

操作	HTTP方法	功能
createShift	POST	在某公司/团队下创建新班次
getShift	GET	通过班次id获取班次信息
listShifts	POST	通过公司/团队/用户/任务id, 时间范围等信息查询对应班次
listWorkerShifts	POST	通过公司/团队/雇员id和时间范围等信息 查询对应班次
deleteShift	DELETE	删除某个班次
updateShift	PUT	更新某班次信息
bulkPublishShifts	POST	批量发布班次



第 5 部分

Dubbo、Spring Cloud和K8s 该如何选型？（上）

微服务公共关注点

DevOps开发者体验

自愈和自动伸缩

调度和发布

调用链监控

配置管理

微服务

Metrics监控

服务发现和LB

日志监控

弹性和容错

API管理

服务安全

IaaS(操作系统, 虚拟化, 硬件/网络/存储)

Dubbo、Spring Cloud和K8s横向比对

	Dubbo	Spring Cloud	K8s
服务发现和LB	ZK/Nacos + Client	Eureka + Ribbon	Service
API网关	NA	Zuul	Ingress
配置管理	Diamond/Nacos	Spring Cloud Config	ConfigMaps/Secrets
容错限流	Sentinel	Hystrix	HealthCheck/Probe/ServiceMesh
日志监控	ELK	ELK	EFK
Metrics监控	Dubbo Admin/Monitor	Actuator/MicroMeter + Prometheus	Heapster+Prometheus
调用链监控	NA	SpringCloud Sleuth/Zipkin	Jaeger/Zipkin



第 6 部分

Dubbo、Spring Cloud和K8s
该如何选型？（中）

Dubbo、Spring Cloud和K8s横向比对

	Dubbo	Spring Cloud	K8s
应用打包	Jar/War	Uber Jar/War	Docker Image/Helm
服务框架	Dubbo RPC	Spring(Boot) REST	框架无关
发布和调度	NA	NA	Scheduler
自动伸缩和自愈	NA	NA	Scheduler/AutoScaler
进程隔离	NA	NA	Docker/Pod
环境管理	NA	NA	Namespace/Auththorization
资源配额	NA	NA	CPU/Mem Limit , Namespace Quotas
流量治理	ZK + Client	NA	ServiceMesh



第 7 部分

Dubbo、Spring Cloud和K8s 该如何选型？（下）

优劣比对

	Dubbo	Spring Cloud	K8s
亮点	阿里背书 成熟稳定 RPC高性能 流量治理	Netflix/Pivotal背书 社区活跃 开发体验好 抽象组件化好	谷歌背书 平台抽象 全面覆盖微服务关注点(发布) 语言栈无关 社区活跃
不足	技术较老 耦合性高 JVM only 国外社区小	JVM only 运行耗资源	偏DevOps和运维 重量复杂 技术门槛高



我的建议

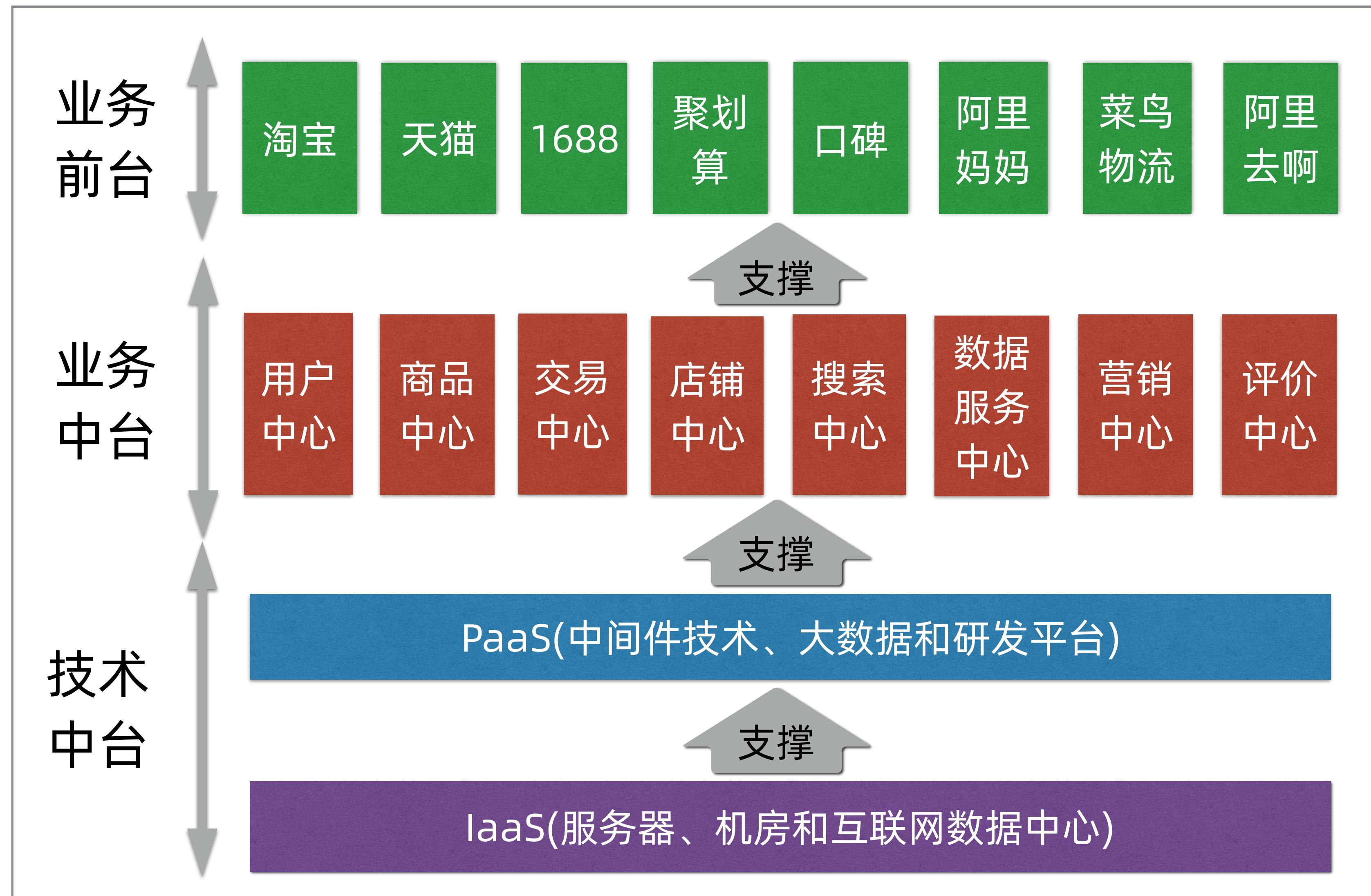
- 理解微服务关注点，根据企业上下文综合考量
- 尽量不要混搭，保持体系一致性
- 个人倾向 K8s + Spring Boot



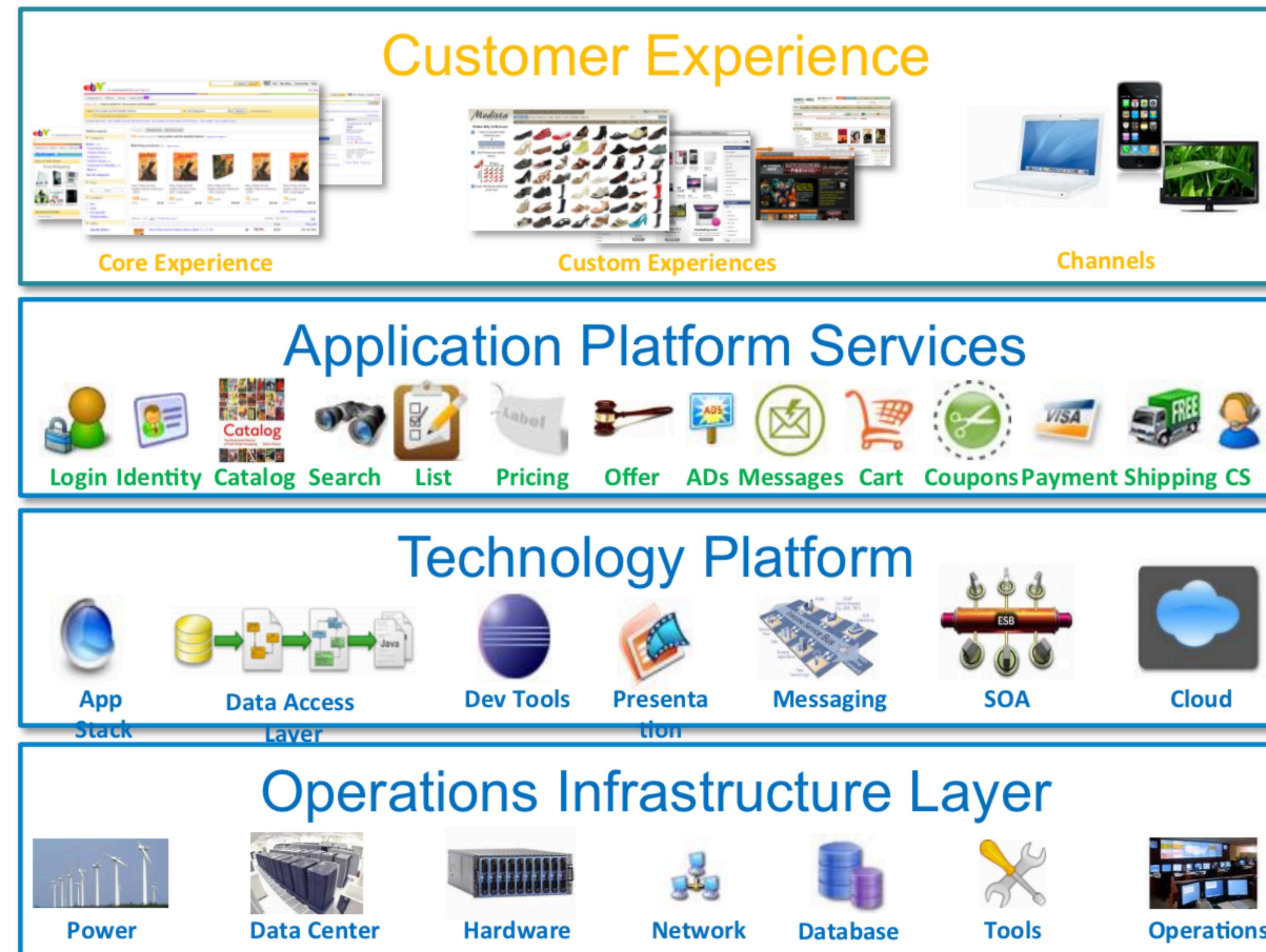
第 8 部分

技术中台到底讲什么？

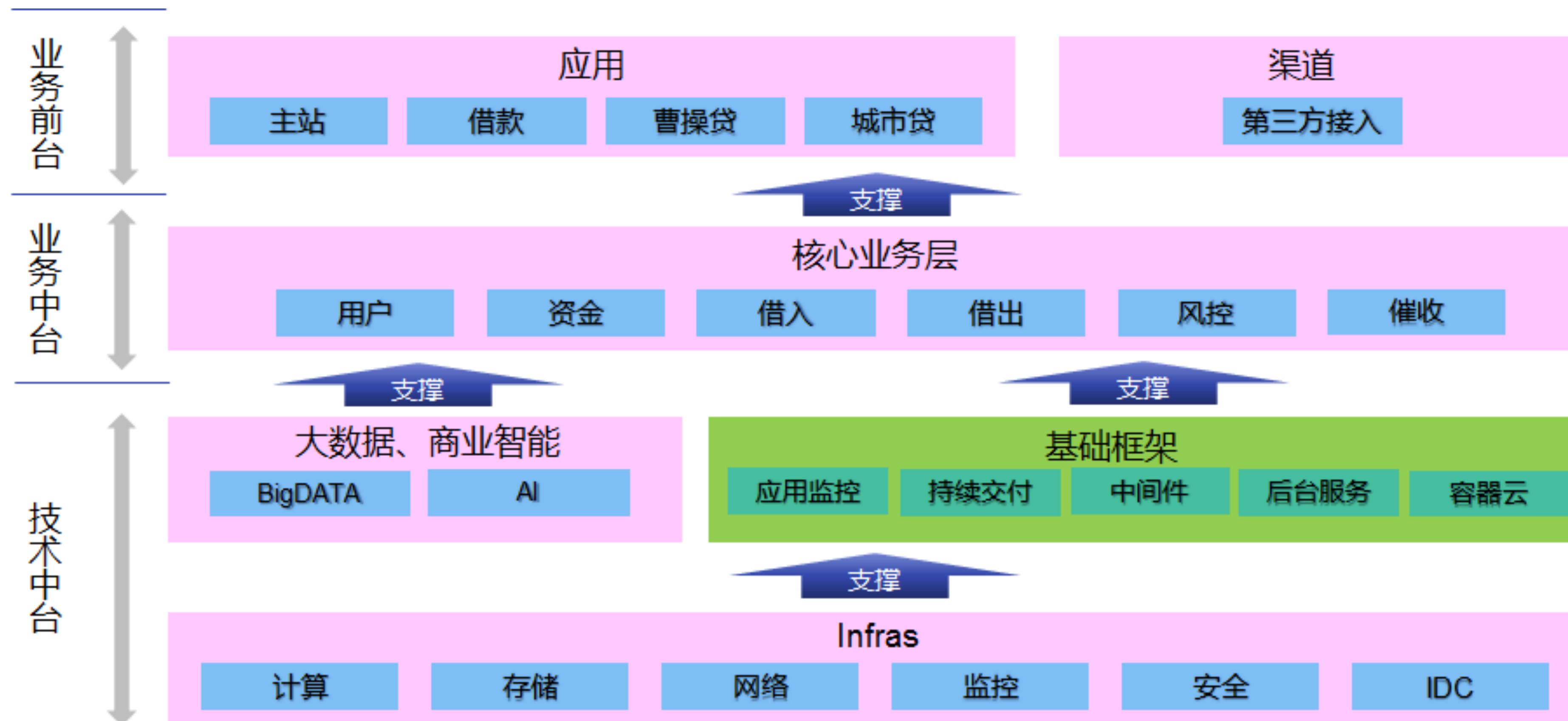
阿里巴巴中台体系



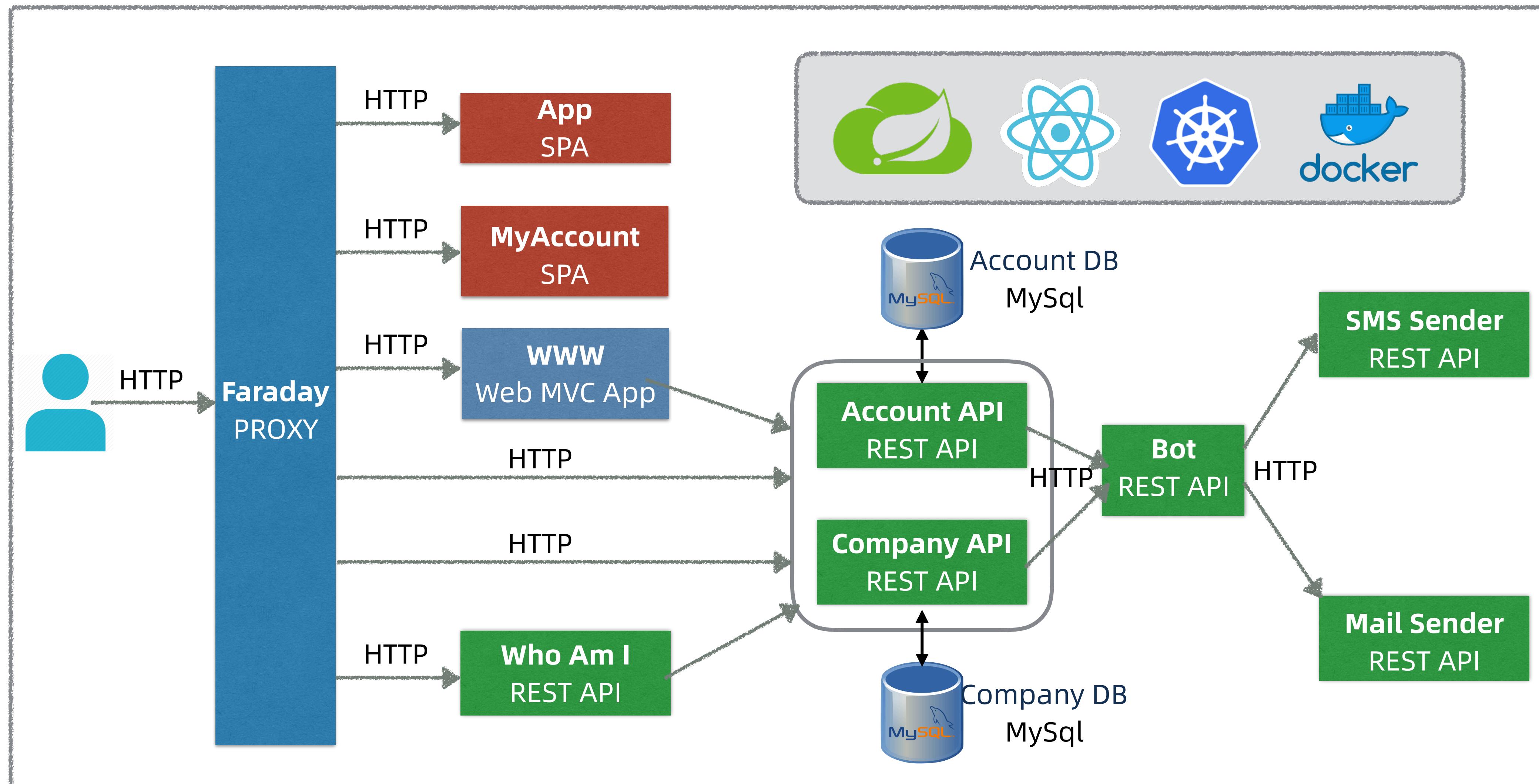
eBay中台架构



拍拍贷中台架构



Staffjoy 的中台



参考链接

1. **Microservice Premium** <https://martinfowler.com/bliki/MicroservicePremium.html>
2. **Monolith First** <https://martinfowler.com/bliki/MonolithFirst.html>
3. 阿里巴巴全面启动中台战略 <https://www.huxiu.com/article/133482/1.html>
4. **The Design Journey of Staffjoy V2** <https://blog.staffjoy.com/staffjoy-v2-ca15ff1a1169>
5. **Axure RP专业原型开发工具** <https://www.axure.com/>



扫码试看/订阅

《Spring Boot & Kubernetes 云原生微服务实践》