

Hyperledger: 助力企业级区块链应用

张海宁 (Henry Zhang)
Chief Architect
VMware China R&D

自我介绍

- VMware中国研发首席架构师，负责云原生应用、区块链、物联网等新兴技术研发
- 开源企业级容器Registry项目Harbor负责人
- 多年全栈工程师
- 《区块链技术指南》、《软件定义存储》作者之一



亨利笔记



《区块链技术指南》



《软件定义存储》

超级账本项目概览

商用区块链的要求

多方共享数据
访问权限控制



用代码描述业务
可验证和签名确认



交易具有合适的可见性
交易需认证身份



多方共同认可交易
满足需求的吞吐量



公有链的不足之处

- 比特币、以太坊等公有链项目，不能满足商用的需求
 - 无保密性(Confidentiality)
 - 无法溯源(Provenance)
 - 确认时间长(Slow confirmation)
 - 无最终性(Finality)
 - 吞吐量低(Throughput)
 - 软件许可(license)
 - 极客主导
- 需要新的解决方案



超级账本项目（Hyperledger）

- Linux基金会于2015年12月成立超级账本项目
- 30个创始成员
 - 科技巨头（IBM、Intel、思科等）
 - 金融大鳄（摩根大通、富国银行、荷兰银行等）
 - 专注区块链的公司（R3, ConsenSys等）
- 目前已经超过110个成员
- 150+ 贡献者
- 8000+ commits

超级账本成员



Premier Member



General Member

超级账本目标

- 基于区块链的企业级分布式账本技术(DLT)
- 用于构建各种行业的商业应用平台
- 模块化、性能和可靠性
- 提供商业友好的许可(Apache V2.0)

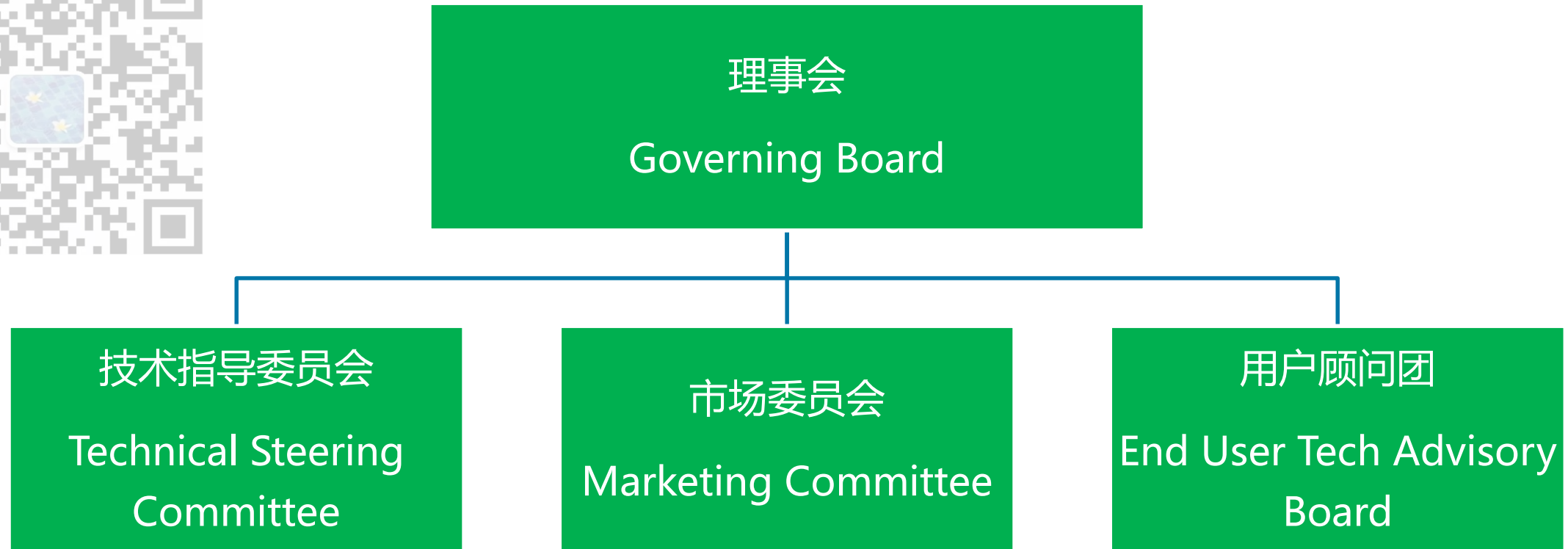
区块链项目对比

	Hyperledger (Fabric)	Bitcoin	Ethereum
项目定位	通用联盟链平台	数字货币系统	通用公有链平台
管理方式	Linux基金会	社区	社区（众筹）
货币	无	BTC 比特币	Ether 以太币
挖矿	无	有	有
状态数据方式	键值数据、文档数据	交易数据	帐号数据
共识网络	PBFT, SBFT等	PoW	PoW, PoS
网络	公开或私有	公开	公开
隐私性	有	无	无
智能合约	Go,Java等多种开发语言	无	Solidity



超级账本项目管理形式

- Premier Member和General Member
- 理事会、技术指导委员会、市场委员会、用户顾问团



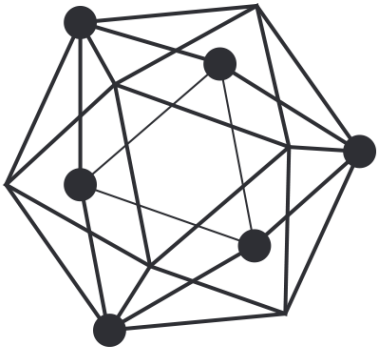
超级账本项目生命周期

- 多个子项目并存
- 每个子项目可有5个阶段

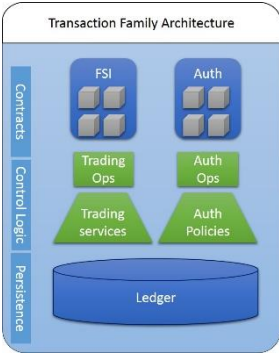


超级账本子项目

- 5个孵化期的子项目



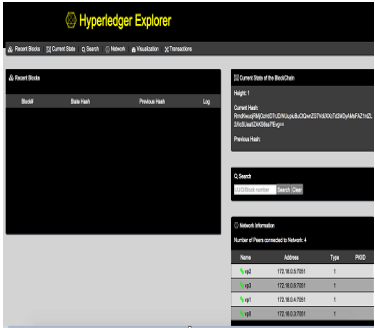
Fabric



Sawtooth Lake



Iroha



Blockchain Explorer



Cello

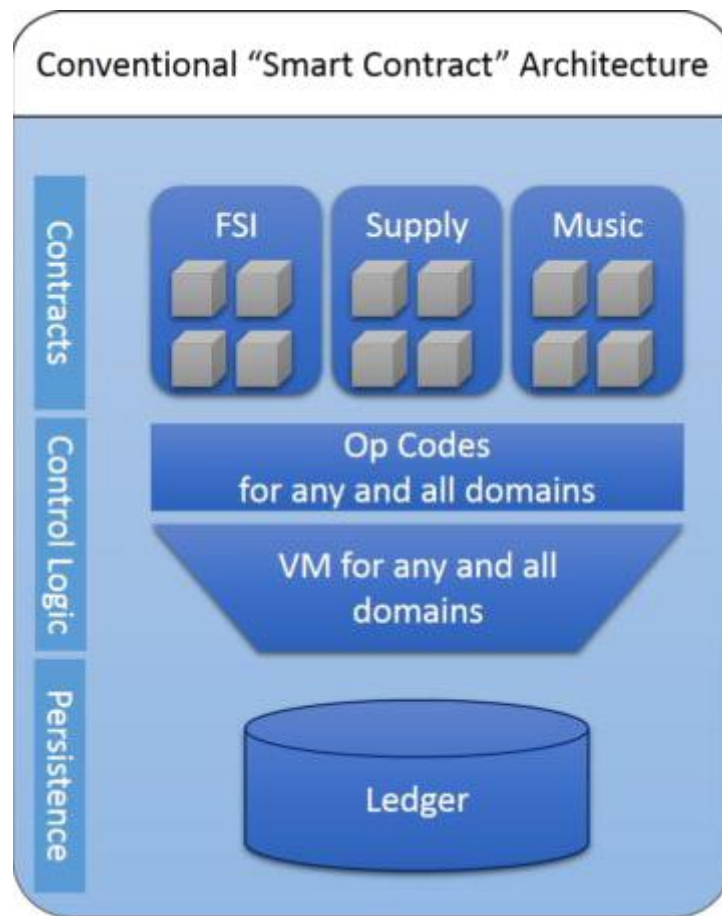
Fabric



- 2015年12月开源
- 主体由IBM的OBC(Open Blockchain)开源代码转化过来
- 增加了DAH和Blockstream两家公司的代码
- 项目以Go语言为主
- 20+贡献者

Sawtooth Lake (锯齿湖)

- 由Intel发起，2016年4月开源
- 项目以Python语言为主
- 20+ 贡献者
- 主要功能
 - PoET共识算法
 - 交易家族



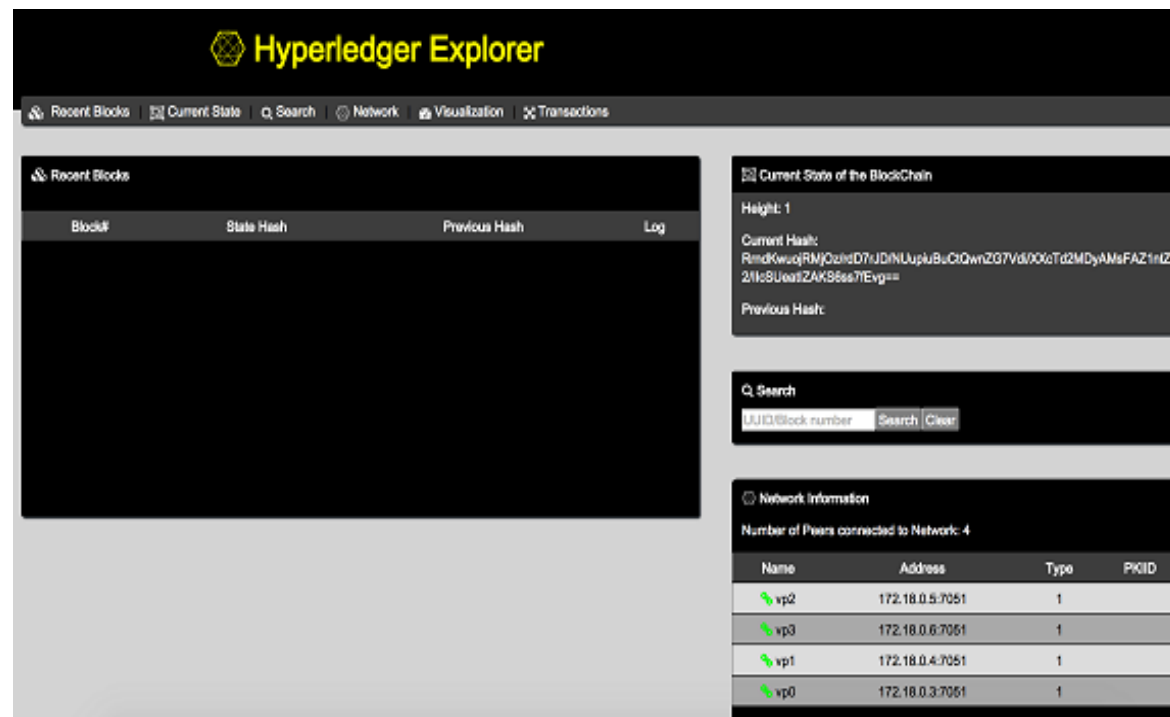
Iroha (色彩)

- 2016年10月开源
- 由日本Soramitsu和日立等发起
- 侧重开发移动应用 (Android, iOS, js)
- 项目以C++语言为主，提供各种工具库
- 10+贡献者



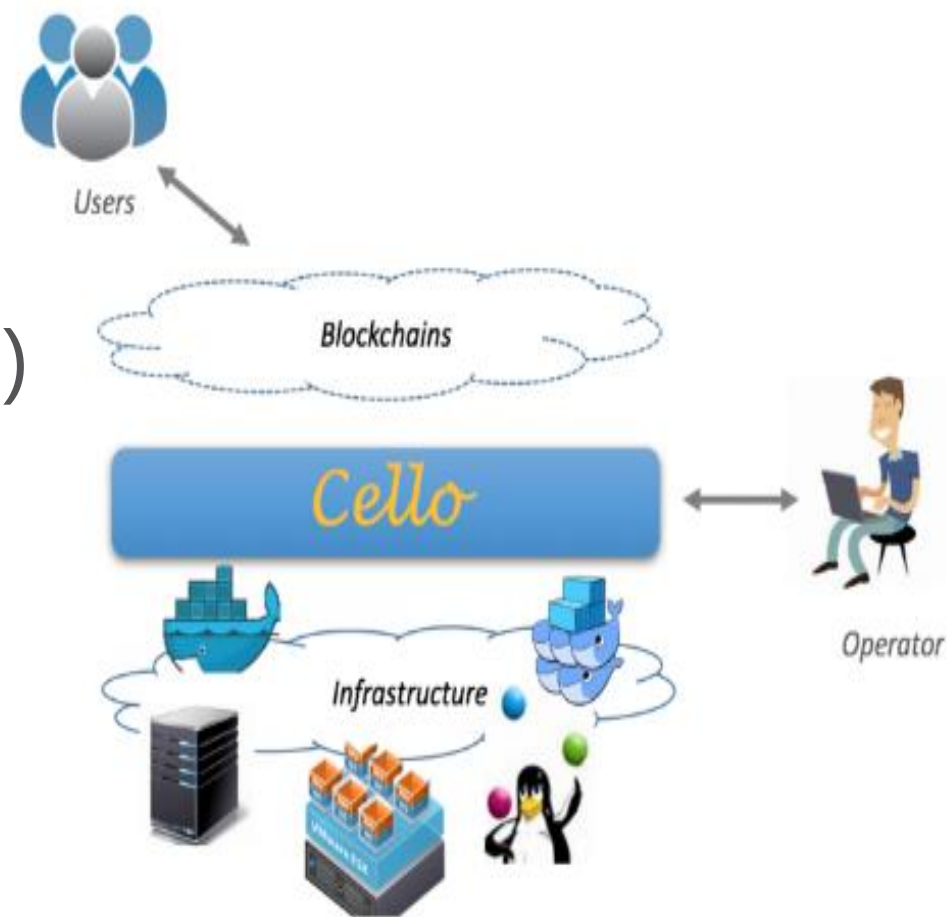
Blockchain Explorer

- 2016年8月开源
- 由Intel, DTCC和IBM等发起
- Blockchain的Web浏览界面
- 项目以Node.js语言为主



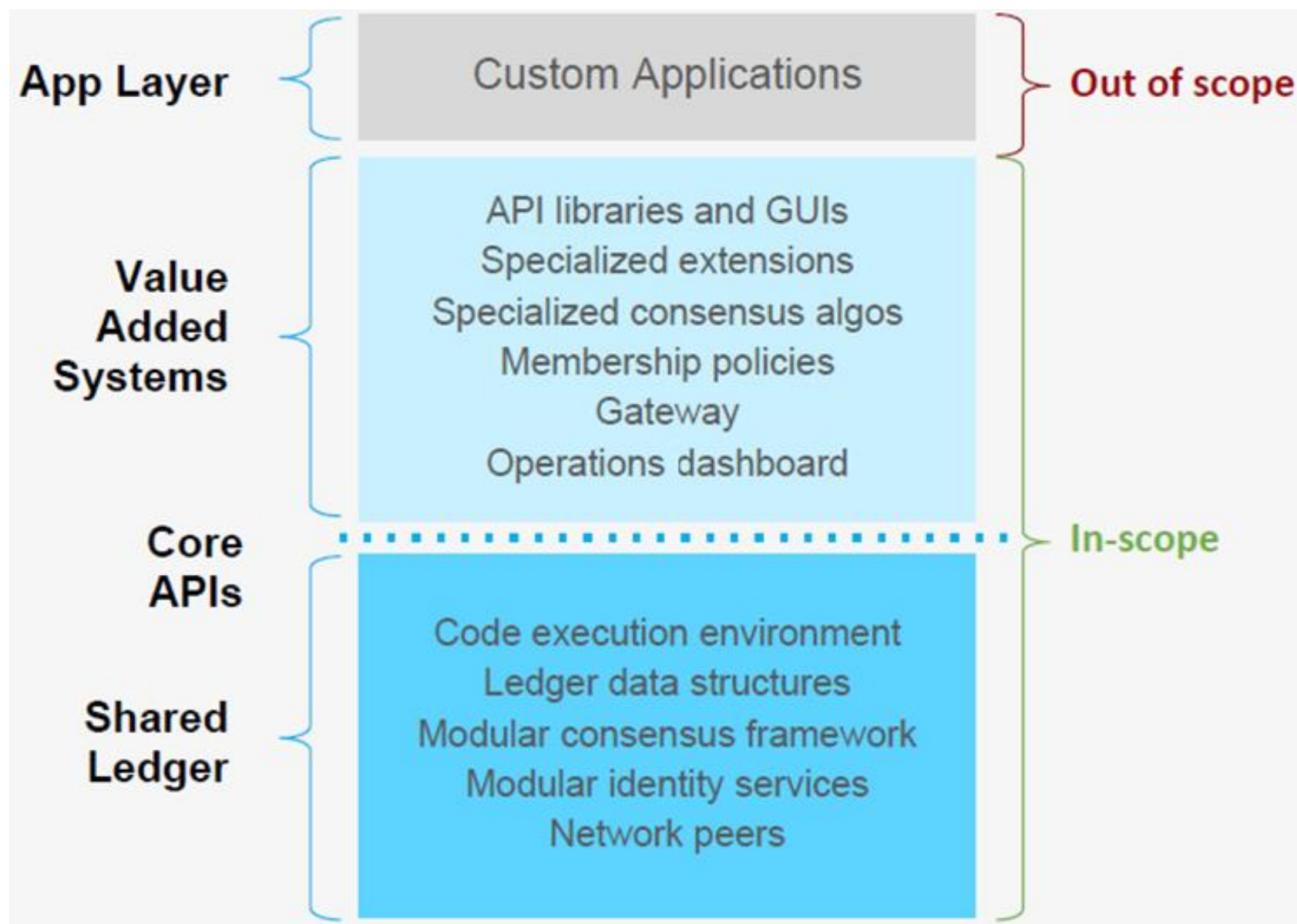
Cello

- 2017年1月开源
- 由IBM发起
- Hyperledger的运维管理工具
 - 提供BaaS (Blockchain as a Service)
 - 多种基础设施的支持



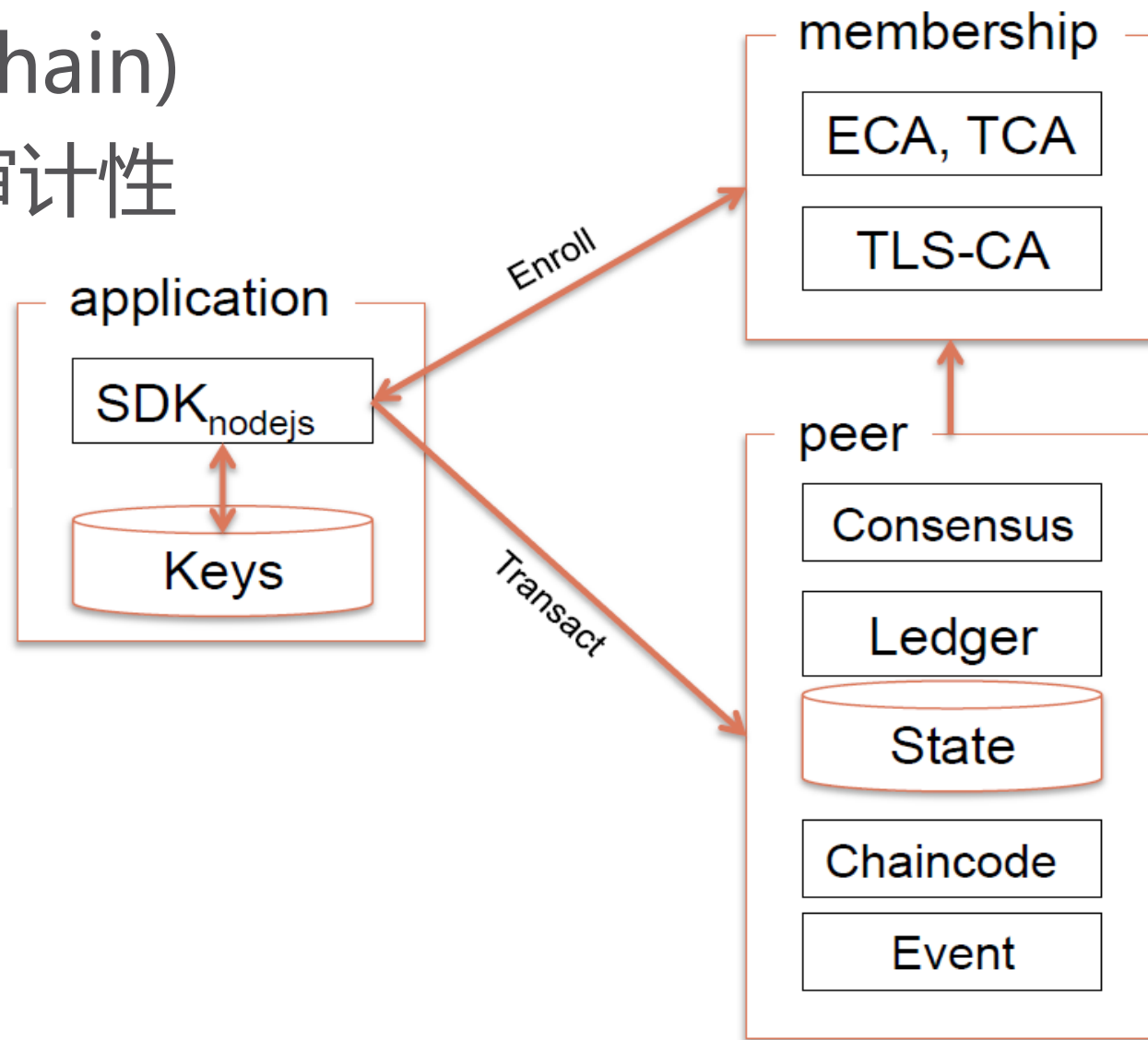
Fabric 子项目

项目覆盖范围

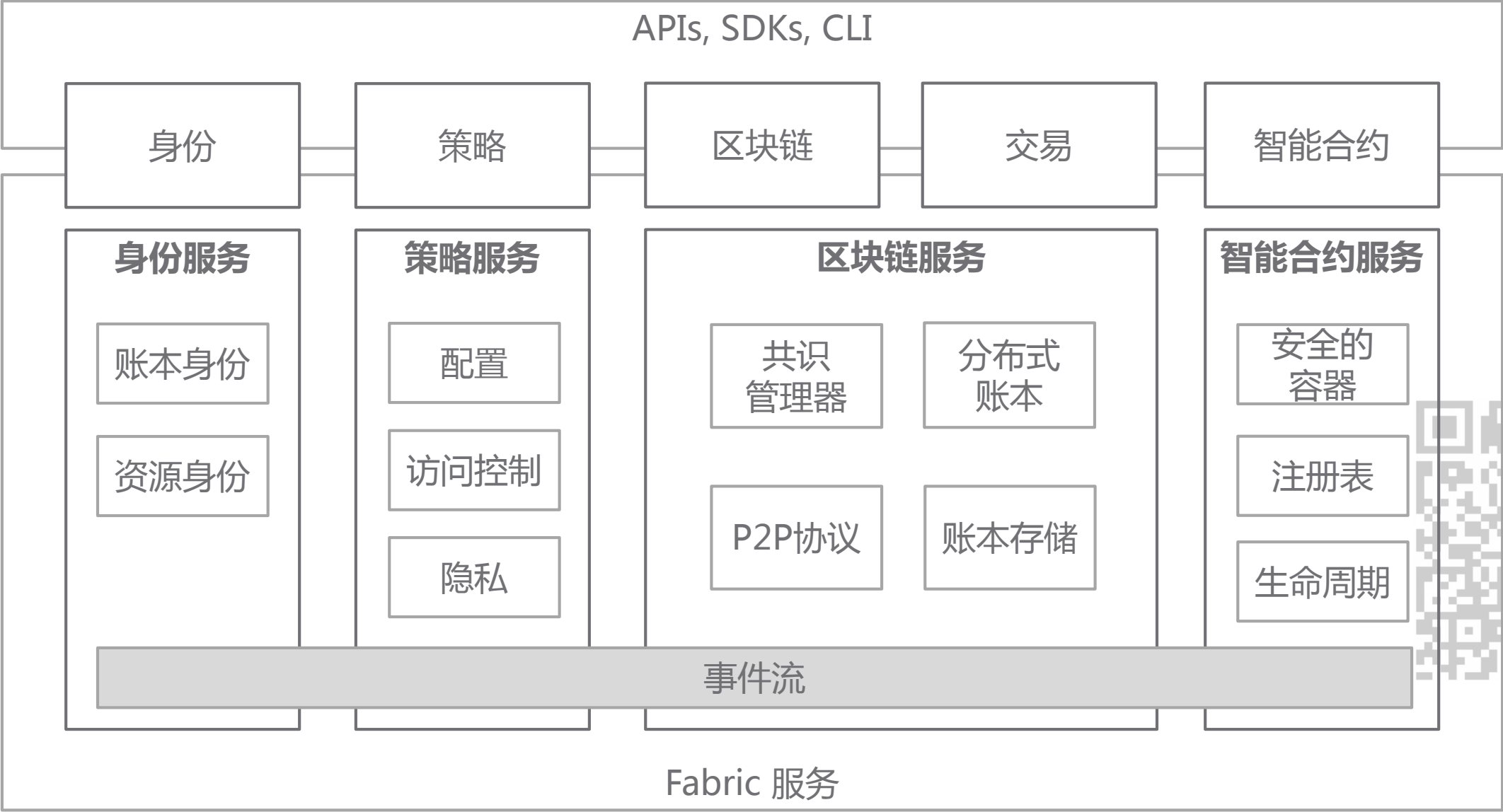


Fabric v0.6 架构 (2016.9)

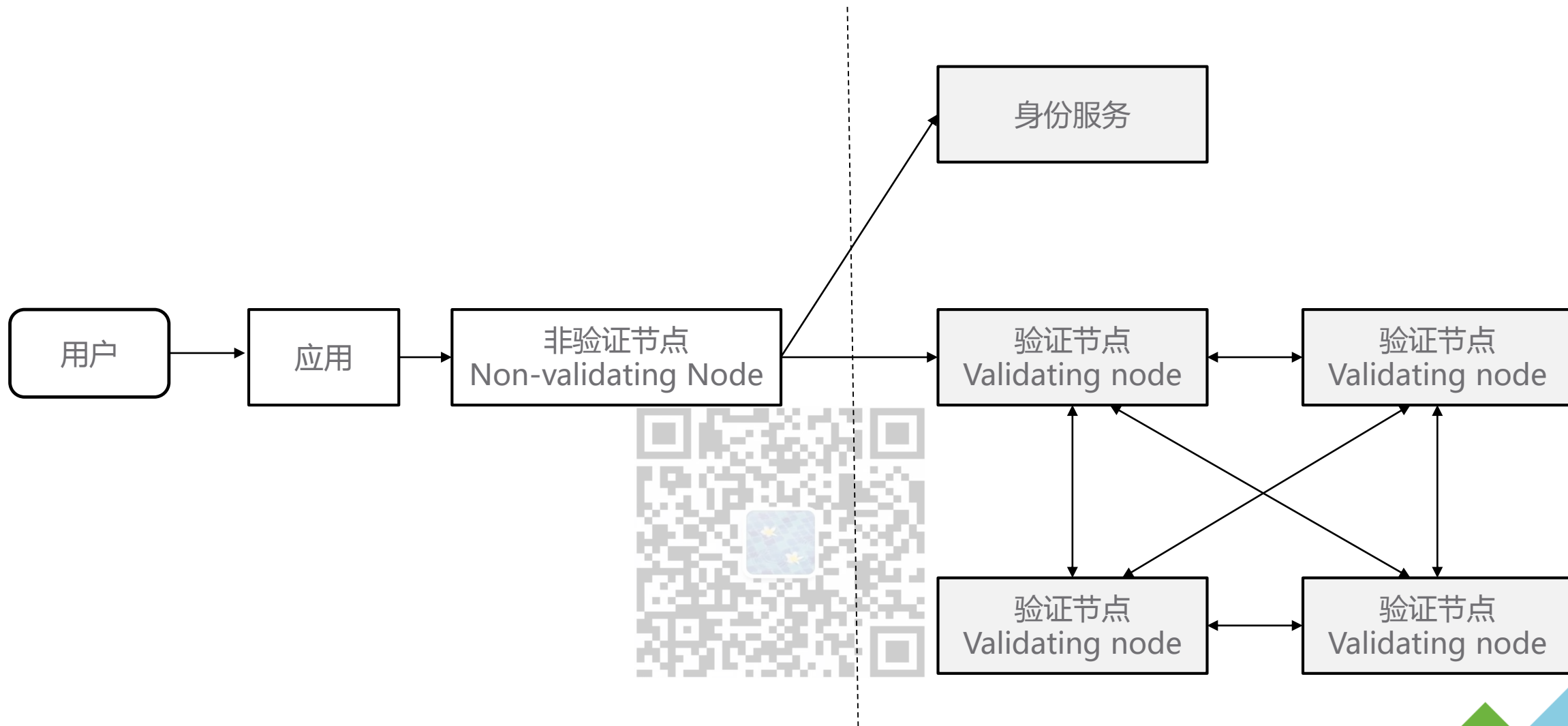
- 许可链方案(Permissioned Chain)
- 基本的隐私性、机密性和可审计性
- 共识算法
 - PBFT
 - SIEVE(原型)
 - NOOPS
- 成员服务



Fabric v0.6 模块结构



Fabric v0.6 部署方式

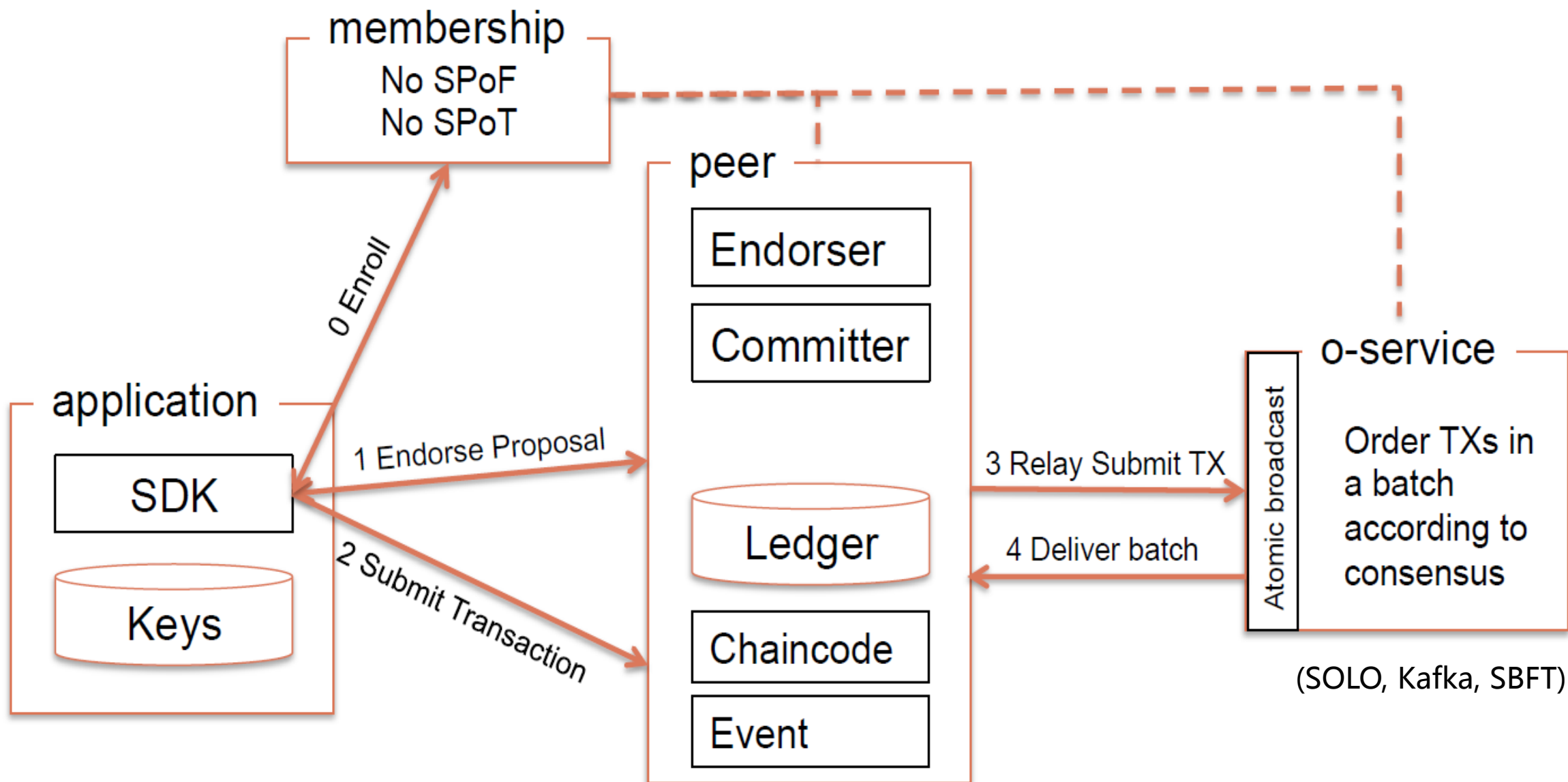


Fabric v0.6的不足

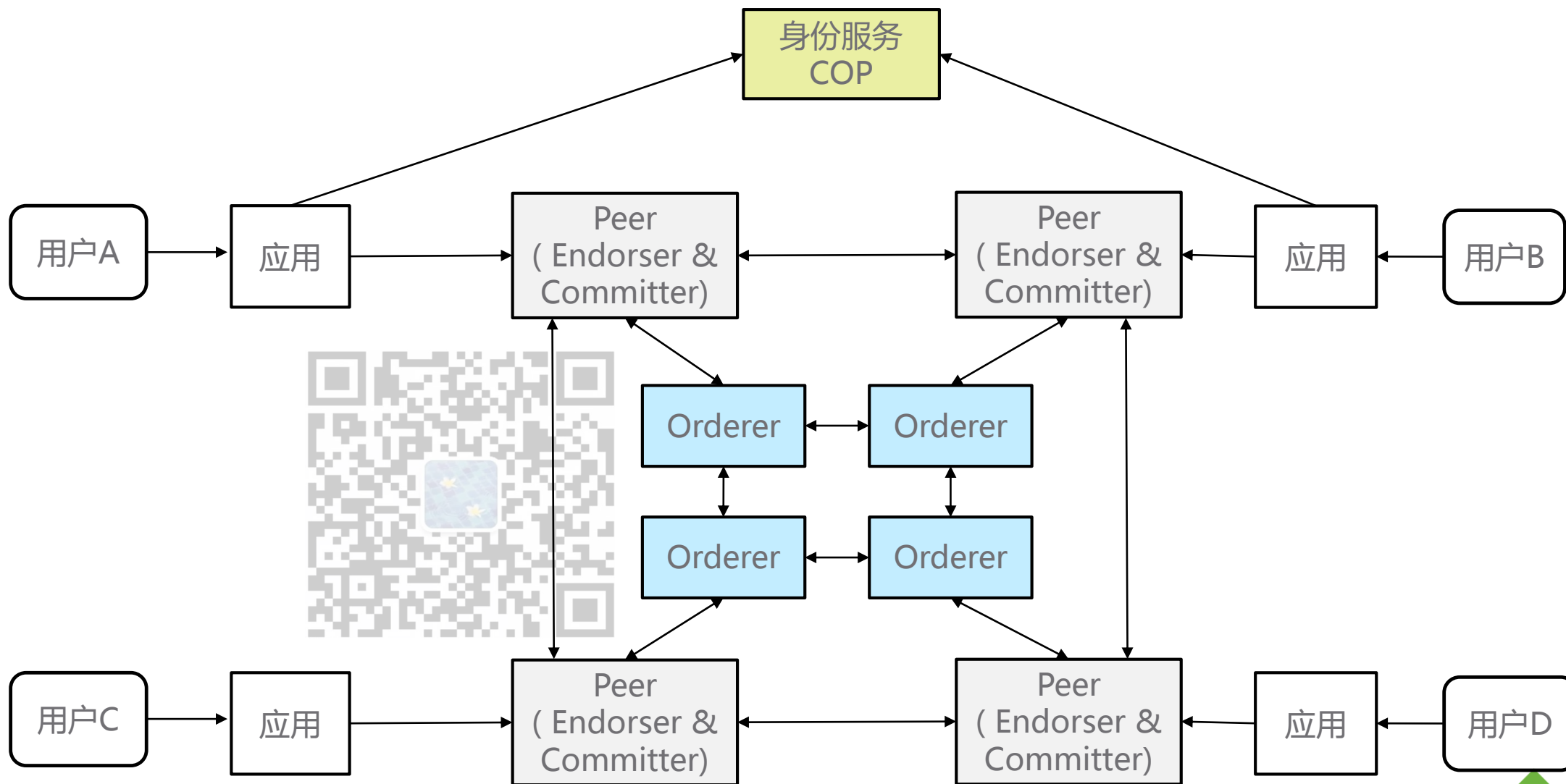
- 成员间的交易没有机密性
- 节点数和吞吐量可扩展性差
- 有不确定性的交易
- 不可升级Fabric和chaincode
- 成员服务是单点故障



Fabric v1.0架构图

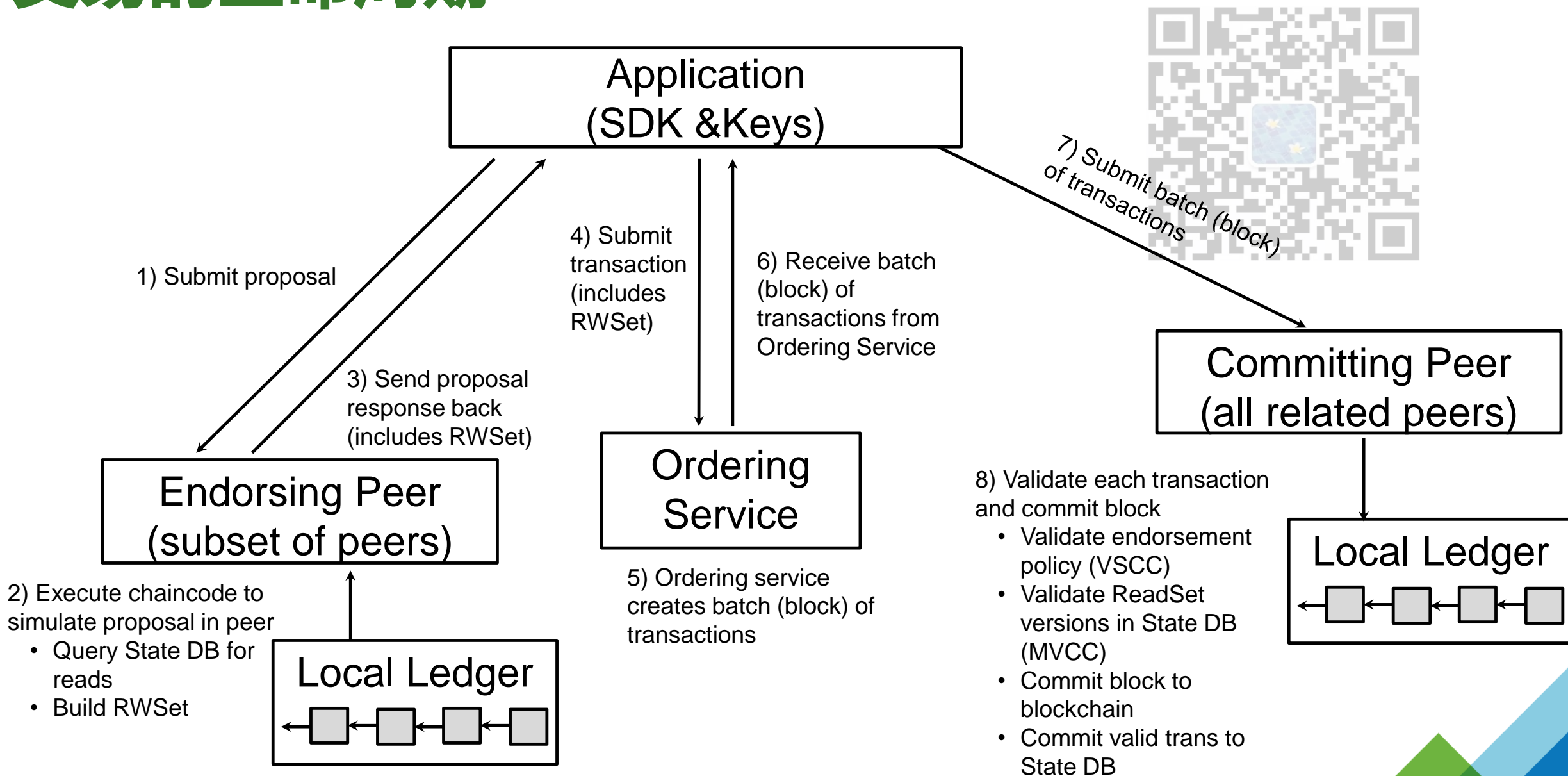


Fabric v1.0部署方式



为简明起见，部分箭头未标注

交易的生命周期

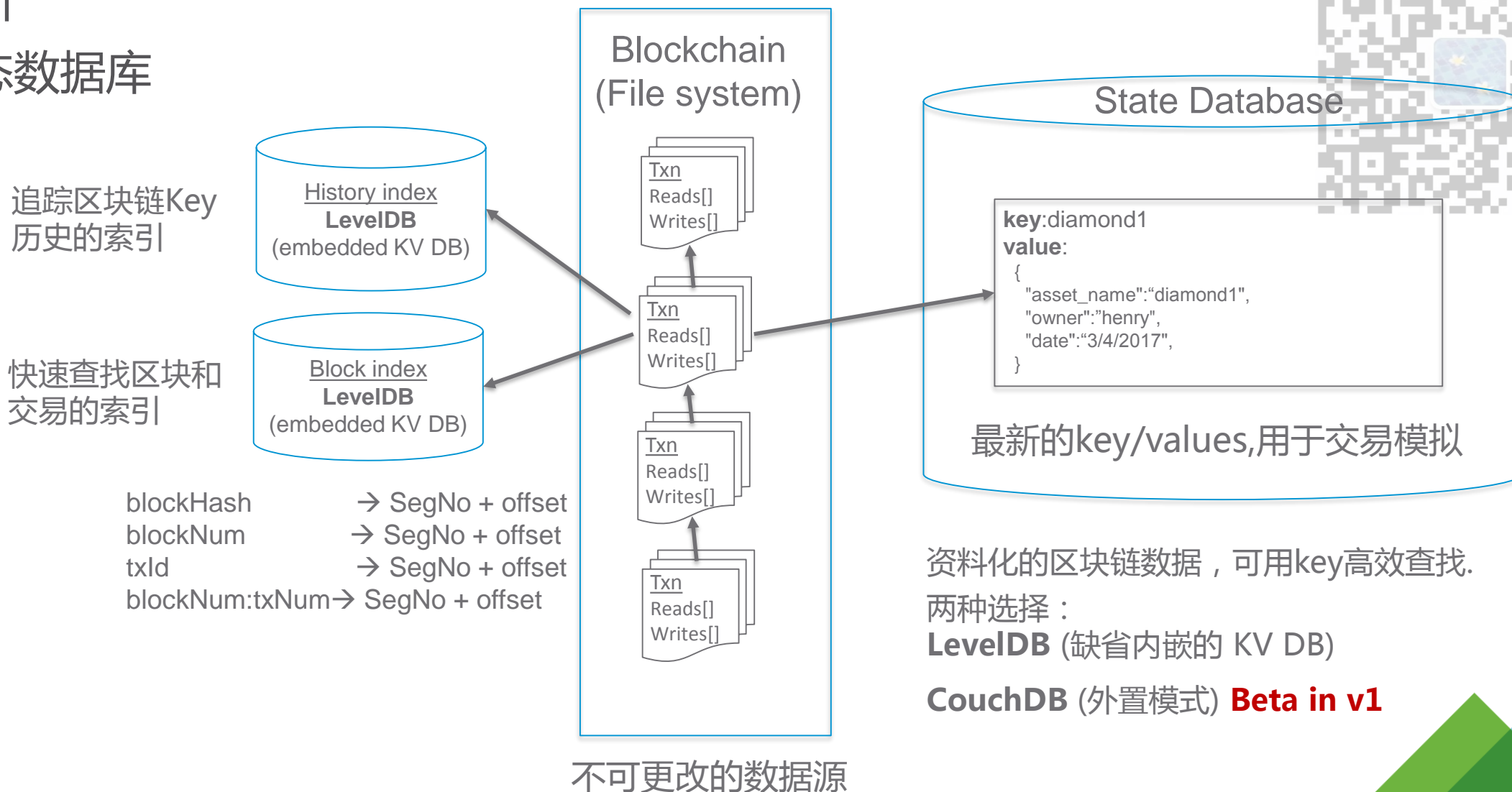


Fabric v1 Transaction Lifecycle

1. Client application creates tran proposal (chaincode function and arguments) and sends to endorsing peer(s).
2. Endorsing peer executes chaincode, generates ReadWriteSet based on keys that were read and written.
3. Endorsing peer(s) send back proposal response (including ReadWriteSet) to client application
4. Client application may or may not submit as a transaction to ordering service. Transaction includes ReadWriteSet from proposal response
5. If client application submitted as transaction, ordering service packages the transaction into a block of ordered transactions.
6. Blocks are delivered to committing peers (including the original endorsing peers).
7. Committing peer
 - runs validation logic (VSCC to check endorsement policy, and MVCC to check that ReadSet versions haven't changed in State DB since simulation time)
 - indicates in block which trans are valid and invalid
 - commits block to blockchain on file system, and commits valid transactions within block to state database 'atomically'
 - fires events so that application client listening via SDK knows which transactions were valid/invalid

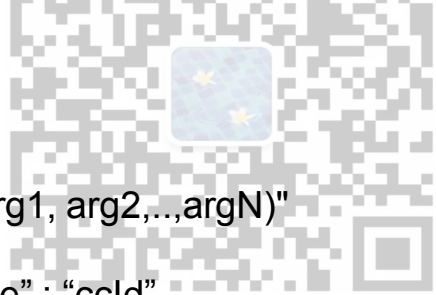
Peer本地账本的数据结构

- 区块链数据
- 索引
- 状态数据库



ReadWriteSet的逻辑结构

- 用于时序校验，解决双花问题 (double spending)
- Endorser
 - 模拟执行交易，生成ReadSet和WriteSet
 - ReadSet是交易前key值的状态
 - WriteSet是交易产生的变化量
- Committer
 - ReadSet作MVCC检查(Multi-Version Concurrency Control)，确保数据没有变化
 - 校验通过后，把交易的WriteSet写入账本和状态数据库



```
Block{
  Transactions [
    {
      "Id" : txUUID2
      "Invoke" : "Method(arg1, arg2,...,argN)"
      "TxRWSet" : [
        { "Chaincode" : "ccId"
          "Reads": [{"key" : "key1", "version" : "v1" }]
          "Writes": [{"key" : "key1", "value" : bytes1}]
        } // end chaincode RWSet
      ] // end TxRWSet
    }, // end transaction with "Id" txUUID2

    { // another transaction },
  ] // end Transactions
} // end Block
```


动态多通道 (Multi-Channel)构造多链系统

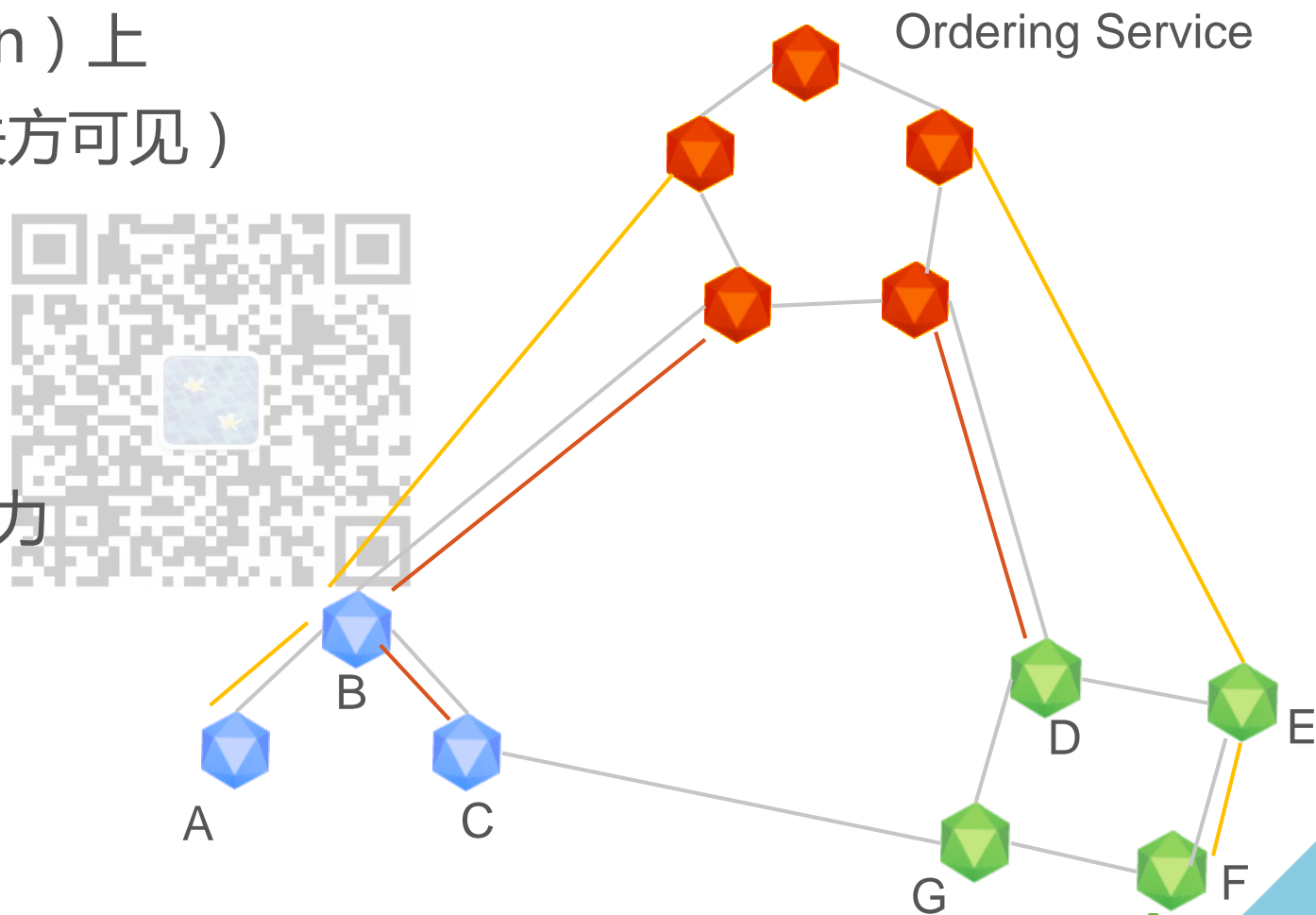
- Chain = Peers + Ledger + Ordering Channel
- Chaincode发布到某个链 (chain) 上
- 链隔离数据和代码 (仅交易相关方可见)
- 每个peer可参与多个链
- 数据可以加密或哈希到其他链中
- 提供隐私性
- 多链并发提高系统的整体吞吐能力

例:

公共链：所有Peer可见 (无隐私)

Chain 1: 包含B,C,D

Chain 2: 包含 A,B, E, F



Fabric 1.0 Roadmap

Endorsement model
Dynamic Multichain
SDK specification
Native SDKs
Pluggable Identity (MSP)
Pluggable Consensus

- Pluggable data-store
- HSM support PKCS11
- Access control
- Upgradable chaincode

Alpha

- Performance
- Security code hardening
- Left-over items

Beta 1

- Continue tuning
- Exit incubator

Beta 2

Release 1.0

2016 Dec

2017 Jan

Feb

March

Fabric 应用场景

适合区块链应用的特征

- 共享数据
- 多个写入者
- 互不信任的成员
- 去中介化
- 交易相关性



- 如果没有上述特征，可能不是适合区块链的用例

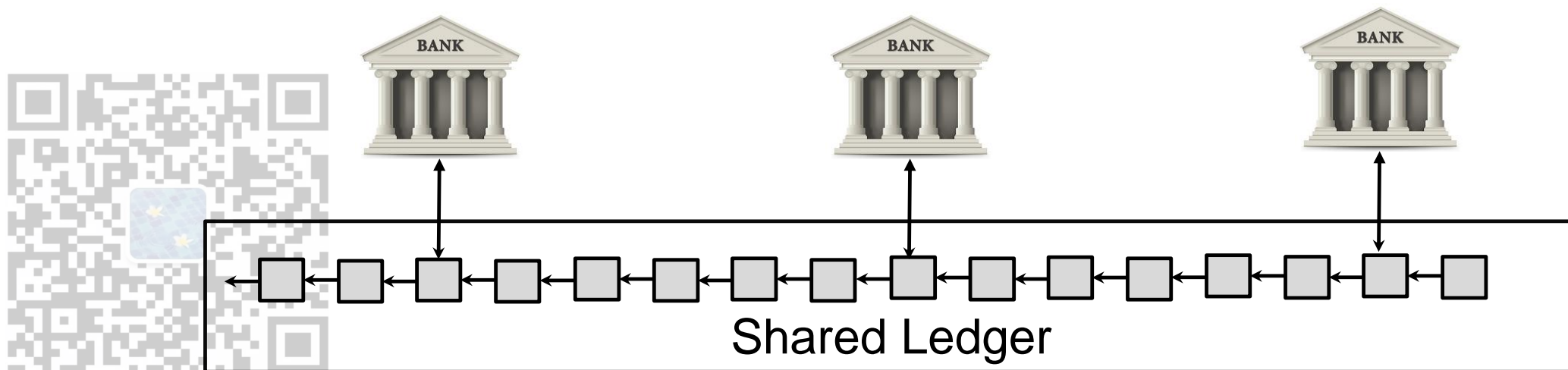


简单例子：银行联行号共享

- 各个银行维护自己的联行号
- 各自写入区块链
- 区块链是联行号的全局视图

The image shows a check from Bank of America. The check number is 1001. The routing and account numbers are highlighted with red boxes and labels below them:

ABA Check Routing Number	Account Number	Check Number	ACH Routing/Transit Number
123456789	000123456789	1001	123456789



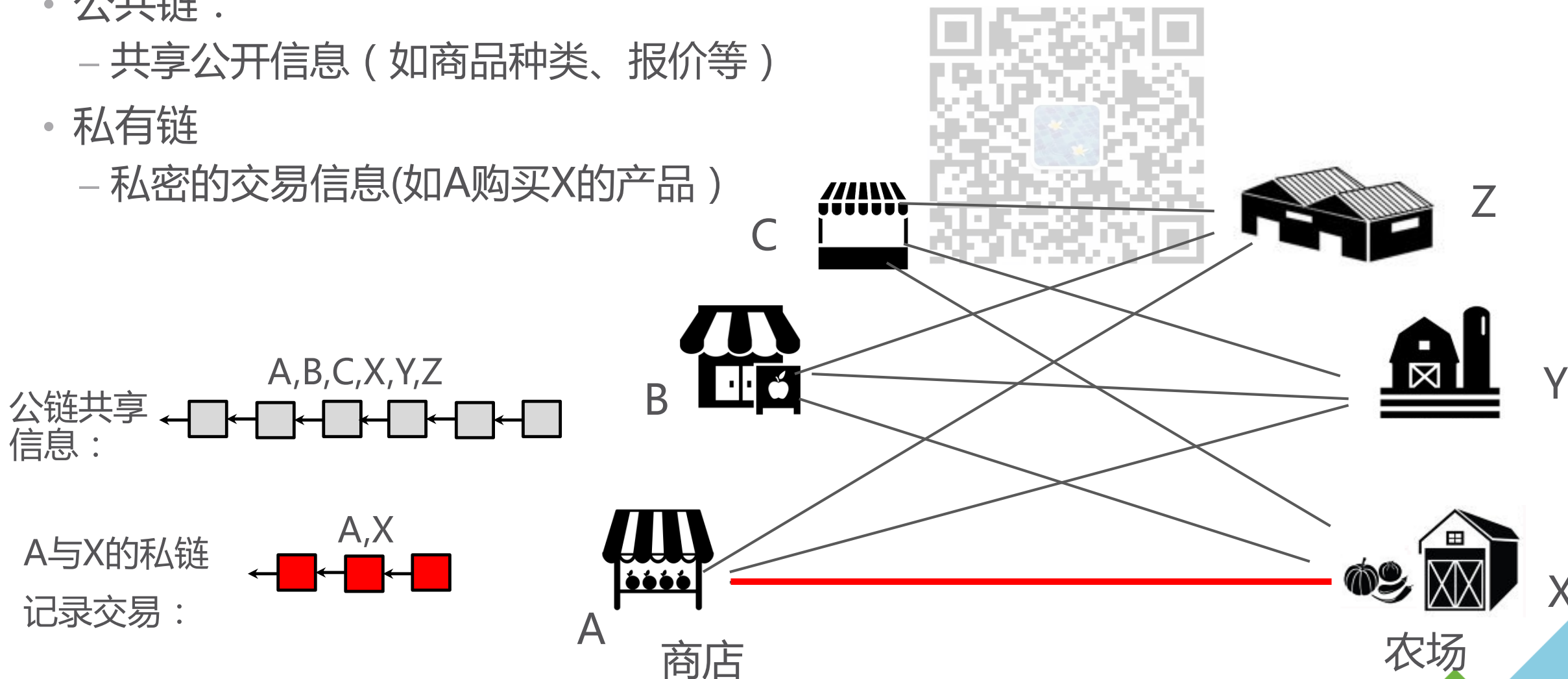
供应链场景 (1)

- 需要共享数据
- 涉及实体很多
- 各自负责更新部分数据
- 成员之间的交易有保密需求



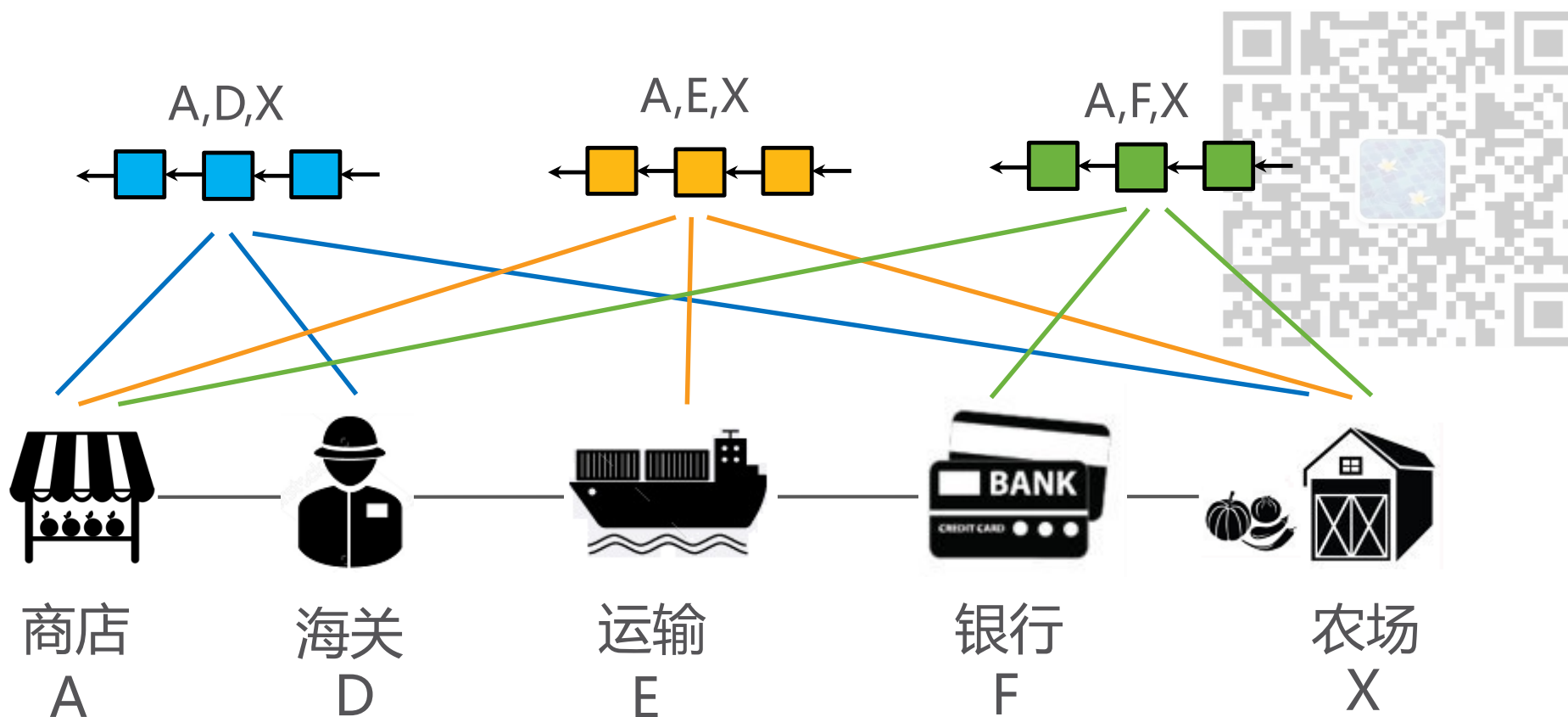
供应链场景(2)

- 公共链：
 - 共享公开信息（如商品种类、报价等）
- 私有链
 - 私密的交易信息(如A购买X的产品)



供应链场景(3)

- 不同群体之间构建不同的私链（和账本）
- 互相独立，分别记账



小结：Hyperledger Fabric

- 提供了交易的机密性
- 权限管理和控制
- 分离了共识和记账职能
- 节点数动态伸缩
- 吞吐量有望提升
- 可升级的智能合约（chaincode）
- 成员服务是高可用



具备了企业级区块链应用的雏形！

Thank you!



亨利笔记

