本文将从以下四个方面分析，替换Fabric的状态库替换为关系型数据可行性，主要分析包括Fabric为什么使用KV型数据库，Fabric数据处理流程、架构设计，及可能存在的问题；接着提出两种可替换的方案，并对比分析两种方案；最后得出分析的结论。

## 总结

为了弥补Fabric现有状态库在模糊查询和复杂查询方面的弱势，从系统灵活性、扩展性、性能和技术实现等方面出发，针对Fabric框架、数据流程设计和现有的研究成果，分析状态库替换成关系型数据库的可行性和可能存在的问题。接着提出了两种替换方案，对比分析方案在技术难度、扩展性和性能方面的优劣。总的来讲，使用外挂关系型数据库的方案有较大优势。但是在具体的实施过程中，数据同步机制的建立和数据格式转换都是替换工作中的核心点。

## 怎么换（改动方案）

### 2.1 两种可选方案

一种方案是，利用Fabric内部数据同步机制，替换节点内部的KV类型的状态库，对Fabric底层代码进行改造，改造内容包括：链码的API、增加结构化SQL查询支持、读写集与结构化数据互相转换等功能，实现状态库的替换。

另一种方案是，外挂关系型数据库，无需对Fabric底层代码改造，通过简单的工具监听排序集群节点的区块分发服务，把收到的区块数据解析成结构化数据，然后同步到一个关系型数据库的集群中，使用关系型数据库集群对外提供服务。

### 2.2 方案对比分析

第一种方案从技术实现上难度较大，成本较高（时间和人工），在扩展性、数据恢复和性能方面都存在较大挑战，并且改造后只是满足通过状态库提供一些模糊和关联查询的需求。此种方案也无法解决系统高可用性的问题，如当有大量的数据量并发操作时，会造成读数据库I/O阻塞，影响正常业务办理工作。

两种方案相比，第二种方案从技术实现和扩展性方面都具有较大优势，只要能够通过一定的技术手段，保证数据同步过程的安全和稳定，即可保证关系型数据库和区块链状态库中数据的一致性，且基于关系型数据库在再对外提供服务，支撑复杂查询和模糊查询的需求，同时分担统计分析时状态库读写压力，提高整个系统可用性。

但是两种方案使用关系型数据作为后端提供服务时，不可避免的问题就是，在业务上线后，需求改动导致数据结构或索引发生变化时，历史和新增数据衔接可能会存在问题。

## 为什么要替换状态库？

### 3.1 Fabric状态库现状

Fabric的状态库基于键值对（k,ver,val）描述数据，其中，键为k,值为val,版本为ver（Height类型，包含区块号和交易序号），它用来存储最新的有效交易执行结果的读写集（实际上只保存了写集合，读集合是为了过滤出有效交易），其状态数据表示指定键的最新值，即“世界状态”。状态库是，例如不动产登记信息写入状态库时，key为不动产单元号+业务号+接入业务编码，val为JSON对象，当转移登记时，不动产单元上一次交易的权利状态需要更新为历史，在查询的时候可根据权利状态过滤出现势的不动产信息。

目前Fabric中状态库的实现只支持Level DB和Couch DB，两者均为KV型数据库。 Level DB是默认的选项，支持基于key的查询，组合键查询和范围查询，当要存储的数据是key-value型时，很适合使用Level DB。此外Level DB嵌入在操作系统中，和网络中的节点结合紧密，较容易做到扩展。Couch DB是一个面向文档的数据库，适合在世界状态值是结构化的数据时使用，例如JSON数据，并且Couch DB支持丰富的查询操作，但是需要自己构造复杂的查询语句。

### 3.2 Fabric为什么用KV型数据库

我们可以设想，在设计一个多组织参与的区块链平台时，使用关系型数据库作为状态库，不同的部门都会有独特的数据模型和控制策略，比如不动产登记机构、银行、税务机构和民政部门，他们之间需要共享数据，就要事先沟通设计共享的数据结构，将各自数据与存储层数据模型对应。最致命的是，系统在运行一段时间后，关系型状态库中数据模型发生了变化，导致需要对存储结构进行更新，并且会对旧的数据产生影响，因此对系统灵活、扩展性方面存在巨大挑战。

所以，Fabric状态库从设计上使用KV型数据库最核心的点就是通过业务和存储的解耦增加可扩展性。通过智能合约（查询层）和KV数据库（存储层)解耦，降低系统的复杂度。解耦之后我们可以发现，智能合约和KV型数据库之间交互，所需的最小方法就是KV的put和get，并保障其作为存储层，满足数据快速读写的需求,这样做其实就是降低的存储层的复杂性，把复杂性移植到了智能合约层，实现查询和存储结构在更多业务场景中的扩展性。

还有一部分考虑是，因为存储层比较复杂，工业化的产品较少，而查询层五花八门，大部分分布式系统都会选择KV型数据库，目的为了保证通用性。从应用界来看，包括阿里、腾讯、华为等提供的区块链平台解决方案，全部使用开源项目Fabric的底层技术框架，并且他们支持的状态库也只有Level DB 和 Couch DB 两种方案，并没有使用关系型数据库替换KV型状态库的方案。

总的来讲，基于系统灵活性、数据存储的扩展性和产品通用性方面的考虑，Fabric使用KV型数据库作为状态库。

### 3.3 Fabric中使用KV型状态库存在问题

在使用Fabric 提供的两种KV型状态库对外提供查询服务时，存在以下的问题，同时也是替换状态库为关系型数据库的需求：

首先，KV型的数据库在模糊和复杂查询能力较弱。在状态库中的值存储的是对象，所以无法像关系型数据库直接通过SQL语句进行具体字段的筛选、模糊匹配和上下数据关联，同样无法快速执行范围查询功能，导致其统计分析功能薄弱：

其次，系统可用性存在威胁。当有数据统计分析的需求时，直接从状态库读取数据，会导致状态库读写数据阻塞，极大降低系统性能，对业务正常办理造成影响。

因此基于以上两点考虑，利用关系型数据库在结构化、模糊查询和统计分析方面的优势，将KV型的状态库替换成关系型数据库，并解决使用状态库进行统计分析时导致的系统压力问题。

## 替换状态库可不可行？

本节将从框架上支不支持,有没有相关的研究成果或应用，分析在扩展性、性能等方面可能会存在的问题。

### 4.1 Fabric设计上支持

**框架上；**

Fabric框架的许多功能支持可拔插，包括共识算法、状态库和密码算法。Fabric从架构设计上将功能模块化进行解耦，实现各个功能模块内的高内聚，功能模块之间的低耦合。Fabric的账本数据存储模块，负责文件系统和键值型数据库的管理，包括账本的创建、区块数据写入、状态数据的写入和更新，通过实现不同的数据库提供者实例，实现状态库替换。

目前Fabric版本支持通过配置文件将状态库更改为Couch DB, 具体操作是通过修改配置文件修改服务地址、用户名和密码，状态库会与区块链网络中的节点保持1:1的对应关系，每创建一个节点就会在节点内部增加一个Couch DB，实现节点复制和系统的高扩展性。

**数据流程上：**

Fabric为了保证数据一致性，在交易预处理阶段、交易排序打包阶段和记账阶段分离。在交易预处理（背书）阶段需要从状态库中读取出数据，并对交易进行模拟执行，生成模拟执行结果（读写集），此时对状态库的操作，并不会影响状态数据；然后，将交易预处理结果交给排序服务进行打包，生成区块；记账阶段，记账节点会验证区块中的交易，并将有效交易的写集合数据更新到状态库。

从数据流程上，可以看出，涉及到状态库读写的阶段包括：背书阶段和记账阶段。

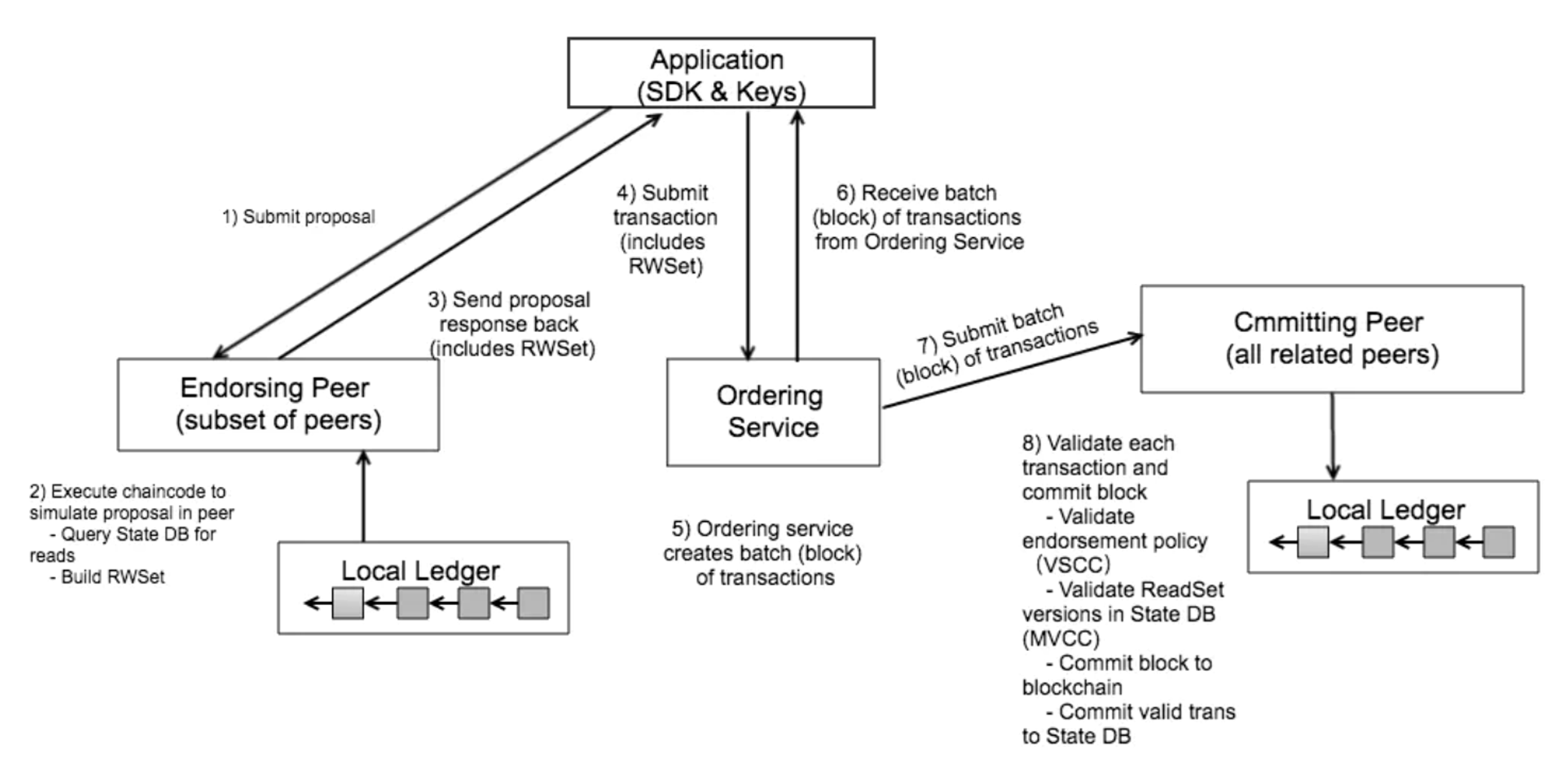


图1. Fabric数据的生命周期

背书阶段，只从状态库中读取数据，并不改变状态库中数据的状态值。记账阶段会改变状态值，但是由于接受的状态统一来自于排序服务，所以保证了所有节点记账的状态值是一致的。

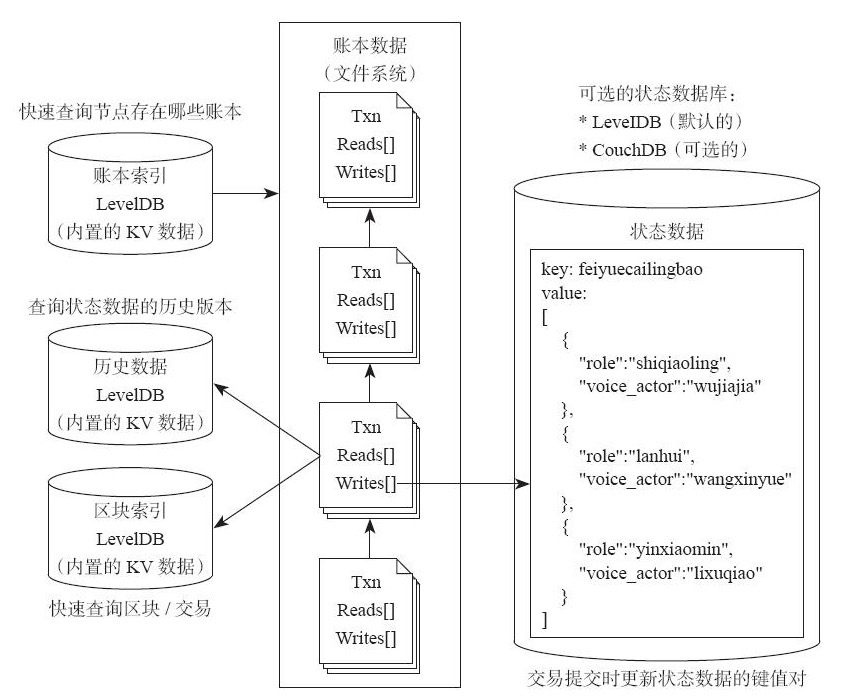


图2. Fabric账本结构

对于将状态库替换成关系型数据库而言，需要在增加对应的数据库创建提供对象，保证每个记账节点和背书节点在运行时能够保持1:1的关系，此外关系型数据库无法直接支持读写集的操作，在智能合约API层需要将读写集转换成关系型数据库中特定的数据结构或策略，即可实现状态数据在关系型数据库的增删改查。

### 4.2 部分研究成果支撑

将Fabric底层状态库替换成关系型数据库，也是Fabric社区研究的一个热点话题，最主要的驱动力来自于五花八门的业务查询需求。IBM相关研究人员自2016年着手研究,并使用关系型数据库PostgreSQL 替代区块链平台底层的账本，他们改造PostgreSQL快照级隔离机制，可在背书阶段并行执行交易，保证互不信任的数据节点间保持数据一致，并对以太坊平台改造，使用并行和串行排序方式，测试复杂合约join操作下的TPS（800左右）。但是由于考虑到系统扩展性和通用性等原因，论文中提出的方案还没有在实际的项目中应用。

### 4.3 替换成关系型数据库可能存在的挑战？

对于Fabric平台，使用关系型数据库作为状态库在扩展性、性能和数据兼容性方面可能存在的问题：

1. 系统的伸缩性，动态扩展性

关系型数据库一般都有固定的表结构，并且需要通过DDL（Data define Language,数据定义语言）语句来修改表结构，数据之间耦合性强，所以不是很容易进行扩展，尤其是需要对数据更新的表做索引或关联变更时，所以不适用于数据模型字段不固定的应用。

1. 写入性能

在有大量数据写入的并发操作时，会导致关系型数据库性能急剧下降，不适用分布式系统这种有大量、小块数据写入的需求；对简单查询需要快速返回时，需要SQL层解析，并带来性能损耗。

1. 无法兼容Fabric底层读写集模式

适配Fabric框架时，至少需要新增中间件，将区块中村存储的交易读写集转换为结构化数据，虽然一些关系型数据库可以直接存储JSON对象，但是这样就丧失了关系型数据库在模糊查询和关联查询的优势。