## 本文将从区块链产生的背景开始介绍比特币热点词汇，探讨区块链发展趋势，并逐步介绍区块链的基本概念与核心技术，进而分析Hyperledger Fabric的架构与核心机制。

## 1．区块链产生背景、发展及涉及的主要技术

### 1.1 背景

#### 1.1.1 区块链的背景

传统互联网上的交易都需要借助可信赖的第三方机构来处理电子支付信息，交易双方信赖第三方机构。

回忆一下你在网上购买一本书的流程：

第一步，你下单并把钱打给支付宝；

第二步，支付宝收款后通知卖家可以发货了；

第三步，卖家收到支付宝通知之后给你发货；

第四步，你收到书之后，觉得满意，在支付宝上选择确认收货；

第五步，支付宝收到通知，把款项打给卖家。流程结束。

你会发现，虽然你是在跟卖家做交易，但是，所有的关键流程都是在跟支付宝打交道。这样的好处在于：万一哪个环节出问题，卖家和买家都可以通过支付宝寻求帮助，让支付宝做出仲裁。这就是一个最简单的基于中心化思维构建的交易模型，它的价值显著，就是建立权威，通过权威背书来获得多方的信任，同时依赖权威方背后的资本和技术实力确保数据的可靠安全。但是第三方介入也有很多缺点：

- 高度中心化系统中单节点故障带来系统失效

- 暴露隐私（第三方为了验证信息，需提供交易双方信息）

试想如果去掉第三方机构，交易双方直接交易，这样怎么保证交易生效呢？区块链技术的出现就是为了解决这样的问题，区块链是一个基于密码学原理建立起点对点信任关系，使得达成一致的双方直接交易，并且公布给所有的见证人。

在2008年中本聪在网络上发表的一篇论文《比特币：一种点对点的电子现金系统》（《Bitcoin: A Peer-to-Peer Electronic Cash System》）,这是比特币和区块链理论形成中重要的奠基性论文，并真正开启了虚拟数字货币与区块链应用的新时代。在中本聪的这篇论文中，他解决了之前发行虚拟数字货币存在的货币伪造、双重支付（或称“双花”, Double Spent）、匿名化交易、中心化货币发行等挑战，可以不依赖第三方信用机构进行背书，无须基于中心化货币发行体系，在全球范围内实现了点对点交易的可靠记账。（**Endorse: 背书**，金融上的意义为：指持票人为将票据权利转让给他人或者将一定的票据权利授予他人行使，而在票据背面或者粘单上记载有关事项并签章的行为。通常我们引申为对某个事情负责。在我们的共识机制的投票环节里，背书意味着参与投票。）

比特币持续11年不间断的正常运行表明，这种分布式架构在适当的激励机制与共识算法的作用下支撑全球范围交易，为当前中心化架构提供了解决思路。因此，在比特币的生态中，每个人都可以参与记账，充分利用非对称加密、哈希算法等现代密码学技术确保比特币交易不可伪造、不可篡改且可溯源的特性，并通过记账奖励的方式激发每个节点参与记账的积极性，这就使得黑客的攻击成本非常高昂，还不如“挖矿”参与社区贡献。正是这种结合正反馈奖励机制与博弈心理学的巧妙设计，使得比特币交易系统迅速发展为一个国际共享的分布式账本系统。

区块链定义，它是一种在对等网络环境下，通过透明和可信规则，（按照时间戳顺序）构建**不可伪造**、**不可篡**改和**可追溯**的块链式数据结构，实现和管理事务（或称交易）处理的模式。实际上，它是一种分布式数据存储、点对点传输、共识机制、加密算法等计算机技术的新型应用模式。

#### 1.1.2 发展现状

根据区块链带来的革新与应用范围将区块链技术分为区块链1.0、区块链2.0和区块链3.0。

（1）区块链1.0：以比特币为代表的可编程“虚拟货币”

中本聪设计的比特币为代表的可信的去中心的电子货币交易系统。

（2）区块链2.0：基于区块链的可编程金融

研究发现，数字货币底层的区块链技术同样可以应用到金融的其他领域，并将比特币中的脚本合约系统重新抽象成 “智能合约”，同时利用区块链特点，用于注册、确认与转移各种不同类型的资产及合约。在此基础上进行探索实践，包括股权登记转让、证券、私募股权、跨境支付、金融衍生品合约等。

智能合约解释：是一套以数字形式定义的承诺，包括合约参与方可以在执行这些承诺的协议，他是一种用计算机语言取代法律语言去记录条款的合约。

（3）区块链3.0：将区块链扩展到其他领域应用

在法律、零售业、物联网、公共医疗、公益事业等领域，依靠区块链提供的信任平台实现信息共享与身份自证明，而不用完全依赖于权威的第三方认证机构，从而降低信用认证成本，提高行业运行效率。

根据实际应用场景和V3.0特点引出Hyper ledger Fabric。

#### 1.1.3 区块链模型分类

根据不同的应用场景与设计体系，区块链可以采用对应的开放共享与节点权限方案，包括如下三种区块链部署模型。

公有链：任意区块链服务客户都可以使用，任意节点均可接入，由所有节点共同参与共识和读写数据，具有较强的去中心化特征，如比特币和以太坊；

联盟链：只有利益相关的特定区块链服务客户才能使用，节点只有经过授权许可后方可接入网络，接入节点按照规则参与共识和读写数据，具有较弱的去中心化的特征，如Hyperledger Fabric；

私有链：仅由单个区块链服务客户使用，仅有授权的节点才能接入，并按照规则参与共识和读写数据。



#### 1.1.4 区块链的应用及意义

目前，区块链技术应用已延伸到数字金融、物联网、智能制造、供应链管理、数字资产交易等多个领域。全球主要国家都在加快布局区块链技术发展。我国在区块链领域拥有良好基础，要加快推动区块链技术和产业创新发展，积极推进区块链和经济社会融合发展。

区块链技术应用的意义：

（1）摆脱了传统信任体系中需要第三方信息验证的信用确认模式；

（2）减少企业联盟间繁冗业务流程的生产成本，有效地降低信用体系构建成本；

（3）为弱信任主体间提供可信平台支持共享敏感数据，提高跨组织体系要素的协同效率；

（4）提高信息监管透明度，避免高度中心化系统中单点故障带来的系统失效，降低黑客攻击风险，提高链上资产的真实性、可信性与安全性。

因此，区块链具有改变未来社会生产与生活形态的潜力，赋能产业经济推动共享经济的普及与发展，催生新的区块链产业与促进相关产业升级，创造社会经济价值。

### 1.2 区块链的数据结构

区块和链

区块链上记录的区块对象是打包了一段时间内发生的交易与状态的集合。交易是指每次改变状态变化的操作，并拥有唯一的交易标识（如哈希值）。同时，区块头部保存了上一个区块的哈希值、当前区块的哈希值等信息，通常以一种链式结构保存所有的区块结构。

区块链具有不可伪造和防篡改、透明可信、可追溯的特点，如何实现不可伪造、不可篡改和可追溯？

Genesis block most recent block

B4

B3

B2

B1

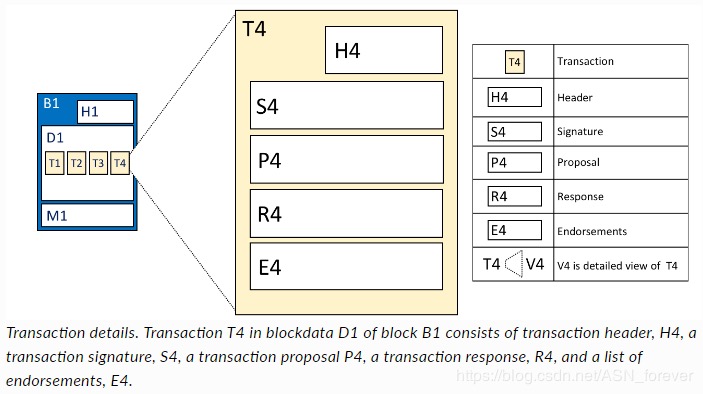
首先区块结构介绍：

区块由三个部分组成，分别是区块头、区块数据和区块元数据。

1.区块头包含三个属性（区块号、当前区块哈希、前一个区块的哈希），当一个区块被创建时写入。

2.区块数据包含的是排序后的交易列表。当区块被ordering service创建时写入。

3.区块元数据包括区块的写入时间，以及区块写入者的证书、公钥和签名。



区块内交易数据结构介绍：

交易头：包含交易的元数据，如链码名称、版本等；

交易签名：包含由客户端应用程序创建的加密签名，作用是判断交易是否被篡改；

交易提案：作用是对由应用程序提供给智能合约的输入参数进行编码。当智能合约运行时，提案负责将参数传递过去；

交易响应：是智能合约的输出，包含的是世界状态在交易前后的值，以读写集的形式展示。

现在我们可以回答之前的提问了，由于每个区块结构内存储的内容有差异，通过Hash散列函数的计算后,每个区块的hash值也是不同的，这样就可以把前一个区块的哈希值作为“链”连接起来，形成一种链式结构保存。如果要修改已经上链的某个区块内交易数据，那么该区块后面所有的区块就需要改动，而且需要将所有分布式节点的数据进行改动。

区块链存储结构本身的不可伪造和防篡改、透明可信、可追溯的特点，决定的了区块链技术最为本质的特点。

### 1.3 涉及主要技术

#### 1.3.1 分布式存储由来及作用

分布式存储就是将大量的普通服务器，通过网络互联，对外作为一个整体提供存储服务。传统的网络存储系统采用集中的存储服务器存放所有数据，存储服务器成为系统性能的瓶颈，也是可靠性和安全性的焦点，不能满足大规模存储应用的需要。分布式网络存储系统采用可扩展的系统结构，利用多台存储服务器分担存储负荷。

分布式存储技术本质上是一种可以在多个网络节点、多个物理地址或者多个组织构成的网络中进行数据分享、同步和复制的数据存储技术，提高了系统的可靠性、可用性和存取效率，还易于扩展。

CAP定理介绍

CAP理论：一个分布式系统不可能同时满足一致性，可用性和分区容错性，必须有所舍弃。

（1）一致性

对于一个将数据副本分布在不同分布式节点上的系统来说，如果对第一个节点的数据进行了更新操作并且更新成功后，却没有是的第二个节点上的数据得到相应的更新，于是在第二个节点上的数据进行读取操作时，获取的依然是旧数据（脏数据），这就是典型的分布式数据不一致的情况。在分布式系统中，如果能够做到针对一个数据项的更新操作执行成功后，所有的用户都可以读取到更新后的值，那么这样的系统就被认为具有严格的一致性（强一致性）。

（2）可用性

可用性是指系统提供的服务必须一直处于可用的状态，对于用户的每一个操作请求总是能够在有限的时间内返回结果。

其中“有限的时间内”是指，对于用户的一个操作请求，系统必须能够在指定的时间内返回对应的处理结果，如果超出这个时间范围，系统则被认为是不可用的。“返回结果”是可用性的一个重要指标，要求系统在完成对用户请求的处理后，返回一个正常的相应结果。正常的响应结果通常能够明确的反应出对请求的处理结果，即成功或失败。

（3）分区容错性

分区容错性约束了一个分布式系统需要具有如下特性：分布式系统在遇到任何网络分区故障的时候，仍然需要能够保证对外提供满足一致性的可用性的服务，除非整个网络环境都发生故障。

分布式存储系统设计时为什么要在CAP定理的三者间取舍？

人们设计大型分布式系统或大型存储系统出发点通常是，他们想获取巨大的性能加成，进而利用数百台计算机的资源来同时完成大量工作。因此，性能问题就成为了最初的诉求。之后，很自然的想法就是将数据分割放到大量的服务器上，这样就可以并行的从多台服务器读取数据。我们将这种方式称之为分片（Sharding）。

Performance ---🡪 Sharding

如果你在成百上千台服务器进行分片，你将会看见常态的故障。如果你有数千台服务器，那么总是会有一台服务器宕机，每天甚至每个小时都可能会发生错误。所以，我们需要自动化的方法而不是人工介入来修复错误。我们需要一个自动的容错系统，这就引出了容错这个话题（fault tolerance）。

Fault ---🡪 Tolerance

实现容错最有用的一种方法是使用复制，只需要维护2-3个数据的副本，当其中一个故障了，你就可以使用另一个。所以，如果想要容错能力，就得有复制（replication）。

Tolerance---🡪 Replication

如果有复制，那就有了两份数据的副本。可以确定的是，如果你不小心，它们就会不一致。所以，你本来设想的是，有了两个数据副本，你可以任意使用其中一个副本来容错。但是如果你不够小心，两个数据的副本就不是完全一致，严格来说，它们就不再互为副本了。而你获取到的数据内容也将取决于你向哪个副本请求数据。这对于应用程序来说就有些麻烦了。所以，如果我们有了复制，我们就有不一致的问题（inconsistency）。

Replication ---🡪 Inconsistency

通过聪明的设计，你可以避免不一致的问题，并且让数据看起来也表现的符合预期。但是为了达到这样的效果，你总是需要额外的工作，需要不同服务器之间通过网络额外的交互，而这样的交互会降低性能。所以如果你想要一致性，你的代价就是低性能。但这明显不是我们最开始所希望的。

Inconsistency ---🡪 Low Performance

当然，这里并不是绝对的。你可以构建性能很高的系统，但是不可避免的，都会陷入到这里的循环来。现实中，如果你想要好的一致性，你就要付出相应的代价。如果你不想付出代价，那就要忍受一些不确定的行为。

#### 1.3.2 共识机制：POW，POS和RAFT

为什么要达成共识？

在分布式系统中，共识就常常应用在这种多副本状态机（Replicated state machines），状态机在每台节点上都存有副本，这些状态机都有相同的初始状态，每次状态转变、下个状态是什么都由相关进程共同决定，每一台节点的日志的值和顺序都相同。每个状态机在“哪个状态是下一个需要处理的状态”这个问题上达成共识，这就是一个共识问题。

共识这个概念，它是指多个服务器在状态上达成一致，但是在一个分布式系统中，因为各种意外可能，有的服务器可能会崩溃或变得不可靠，它就不能和其他服务器达成一致状态。这样就需要一种Consensus协议，一致性协议是为了确保一致性和容错性，也就是即使系统中有一两个服务器宕机，也不会影响系统的可用性。

共识机制的本质就是通过投票或其它大家认可的方式来决定哪个节点，在一定的时间范围内具有打包区块的权利。常见的共识机制有POW、POS和RAFT。

POW(Proof of Work)工作量证明：就是通过计算机对一个数学公式进行计算，谁先计算出的结果与答案结果一样，谁就获得本次记账权。

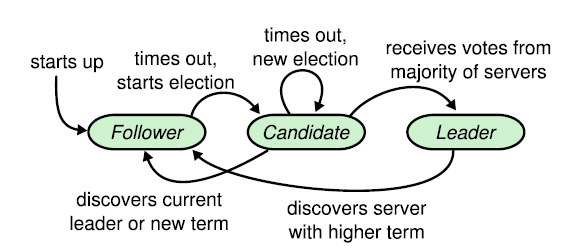
优点：算法简单粗暴，节点无需交换额外信息即可以达成共识，破坏难度大。

缺点：挖矿造成大量的资源浪费；共识达成的周期较长；不适合商业应用。

POS(Proof of Stake)：权益证明，Pow的一种升级共识机制，根据每个节点所占代币的比例和时间，等比例的降低挖矿难度，从而加快找随机数的速度。

优点：在一定程度上缩短了共识达成的时间，节省了资源。

缺点：还是需要挖矿，资源浪费，无法商用。

RAFT算法：

Raft算法通过选举一个高贵的领导人，然后给予他全部的管理复制日志的责任来实现一致性，领导人发起更新数据请求给所有的非领导节点，进行日志复制，进而同步数据。RAFT核心思想是，如果分布式的数据库中初始状态一致，只要之后进行的操作一致就能保证之后的数据一致。

优点：高效1000-1万笔/秒，节点通信无须验证身份签名；容纳故障节点(CFT)。

缺点：只适用于permissioned systems (私有链)，只能容忍故障节点(CFT)，不能容纳恶意节点。

#### 1.3.3 安全机制:哈希函数，非对称加密

区块链中采用了现代密码学中的哈希算法、非对称加密算法等来保证数据机密性、完整性、抗抵赖性等安全特性。

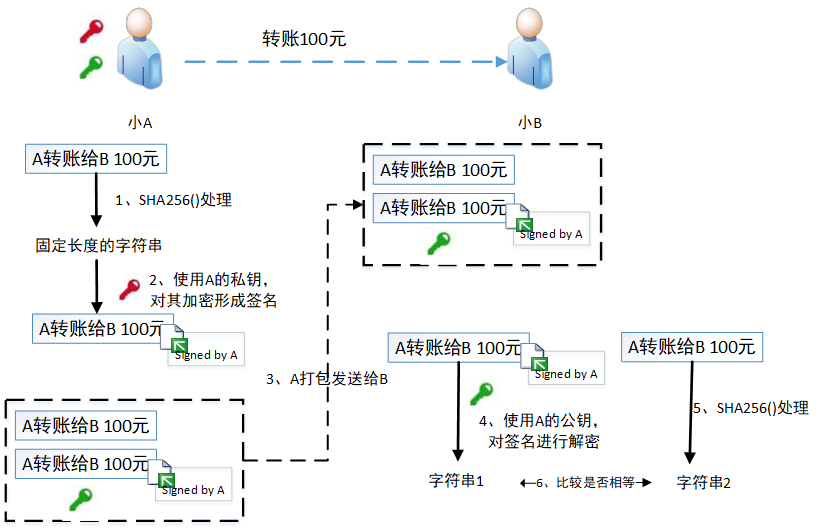
（1）Hash函数

哈希算法可以将任意长度的消息明文转换映射为固定长度的二进制串输出，称为哈希值或散列值，又称为该消息的指纹（fingerprint）或摘要（digest）。即使两个消息只差一个字符，在经过哈希函数计算之后，它们所产生的字符串也会十分杂乱随机且完全不存在任何关联性，被广泛应用于检测签名有效性与完整性的场景中。目前，常见的哈希算法包括MD4（RFC1320，输出128位哈希值）、MD5（RFC1321）、SHA-1（输出长度为160位哈希值）、SHA-2（包括SHA-224、SHA-256、SHA-384、SHA-512等算法）、SM3（国密安全算法）等。其中，MD4已经被证明不够安全，MD5与SHA-1不具备强抗碰撞性，具有安全隐患，因此都不推荐使用。安全术语“碰撞”是指两个不同的消息使用同一个哈希函数计算时获得相同哈希值的情况，哈希算法具有抗碰撞性意味着找到两个能够产生碰撞的消息在计算上是非常困难的（注意不是不可能的）。哈希算法应当具有抗碰撞性、原像不可逆性、难题友好性等特点。目前，区块链通常采用SHA-256（美国国家安全局发明，Secure Hash Algorithm）、Keccak-256CSHA3或更高安全级别的主流算法，常用于构造货币地址、哈希指针、消息摘要等对象。

（2）非对称加密

我们只需要了解，任何人手里都有两把钥匙，其中一把只有自己知道，叫做“私钥”，以及一把可以公布于众，叫做“公钥”；通过私钥加密的信息，必须通过公钥才能解密。

有了这些知识点，我们来看下内部是如何工作的吧，这里拿“A 转账给了B 100元钱” 举例：



第一步：小A会先用SHA256()对自己的小纸条进行处理，得到一个固定长度的字符串，这个字符串就等价于这张小纸条。

第二步：小A使用只有自己知道的那一把私钥，对上面固定长度的字符串进行再加密，生成一份名叫数字签名的字符串，这份数字签名能够充分证明是基于这张小纸条的。你可以这么理解，在现实中，你需要对某一份合同的签署，万一有人拿你曾经在其他地方留下的签名复制粘贴过来怎么办？！最好的办法，就是在你每一次签名的时候，故意在字迹当中留下一些同这份合同存在某种信息关联的小细节，通过对小细节的观察可以知道这个签名有没有被移花接木。步骤一和步骤二的结合就是为了生成这样一份有且仅针对这条小纸条有效的签名。

第三步：小A将「明文的小纸条」、刚刚加密成功的「数字签名」，以及自己那把可以公布于众的「公钥」打包一起发给小B。

第四步：当小B收这三样东西，首先会将明文的小纸条进行SHA256()处理，得到一个字符串，我们将其命名为“字符串2”。然后，小B使用小A公布的公钥，对发过来的数字签名进行解密，得到另外一个“字符串1”。通过比对“字符串1”和“字符串2”的一致性，便可充分证明：小B接受到的小纸条就是小A发出来的小纸条，这张小纸条在中途没有被其他人所篡改；且这张小纸条确实是由小A所编辑。

非对称加密算法适用于签名与验签场景，同时解决了密钥分发的安全性问题，但不适合大量数据的快速加解密过程，如签名消息时通常要先计算消息摘要，再使用私钥对摘要签名（Elliptic Curve Digital Signature Algorithm ECDSA椭圆曲线数字签名算法）。

## 2．Hyper ledger Fabric 基本概念和框架

### 2.1 介绍Hyper ledger Fabric 是什么

一款开源的许可区块链（Permissioned Chian）分布式账本平台。

Hype rledger Fabric 是分布式账本解决方案的平台，采用模块化架构，提供高安全性、弹性、灵活性和可扩展性。它被设计为支持以可插拔方式实现不同组件，并适应复杂的经济生态系统。

Hyperledger Fabric与其他区块链系统最大的不同体现在私有和许可。与开放无需许可的网络系统允许未知身份的参与者加入网络不同（需要通过工作量证明协议来保证交易有效并维护网络的安全），Hyperledger Fabric通过Membership Service Provider(MSP)来登记所有的成员。

Fabric的优点：

避免了比特币与以太坊类公链平台交易效率低下、缺乏完善的身份认证模块等不足，能广泛应用于金融资产存管、供应链、共享经济等领域。

### 2.2 基本概念

介绍Hyper ledger Fabric的基本概念，包括Peer节点、Orderer排序节点、Client客户端等。

**1、Peer节点**

在Hyperledger Fabric网络中，Peer节点指提供交易背书、交易验证、提交账本等服务功能的逻辑节点，包括Endorser背书节点、Committer记账节点等。推荐对性能有严格要求的系统将Peer节点部署在物理节点上。类似于P2P网络，Fabric中每个Peer节点的功能地位都是对等的，它们之间通过服务分工协作以响应来自Fabric客户端（包括CLI命令行客户端和多种语言SDK客户端）的交易请求消息，并共同维护Fabric分布式账本的数据一致性。目前，Fabric提供了如下两种功能角色的Peer节点，其中：

Endorser背书节点：负责接收来自客户端的签名提案消息请求，检查消息后模拟执行交易提案，并对模拟执行结果签名背书，即使用私钥对请求提案、状态变更（读写集）等签名，表示Endorser背书节点认为此次交易是合法有效的，然后将签名背书信息等打包成提案响应消息回复给客户端；

Committer记账节点：负责检查交易消息结构的完整性与合法性、调用VSCC验证交易背书策略、执行MVCC检查读写集冲突等，标记交易的有效性并提交账本，更新本地账本数据库与文件，包括区块数据文件、隐私数据库、区块索引数据库、状态数据库、历史数据库等。

**2、Orderer排序节点**

Orderer排序节点同样属于逻辑节点，负责提供交易广播服务、共识排序服务、区块分发服务等。

交易广播服务：接收与处理交易消息请求（普通交易消息与配置交易消息），过滤检查后提交共识组件；

共识排序服务：共识组件使用共识算法对交易进行排序，并添加到本地待处理的缓存交易消息列表，按照约定的交易出块规则（如配置出块时间、配置出块字节数限制、通道配置消息单独出块等）切割打包成新区块，再保存到本地账本的区块数据文件中；

区块分发服务：接收与处理区块请求消息，从本地账本中获取请求范围内的区块数据回复给请求节点。

因此，Orderer节点在整个Fabric系统中属于核心功能模块，其处理交易排序达成共识的性能将直接影响到整个Fabric系统的出块效率。

**3、Client（客户端）**

客户端（Client）是用户与Fabric网络组件发送请求进行交互的接口，包括如下两种客户端。

Fabric-CA客户端：负责身份鉴别与权限控制功能，包括节点注册登记，身份注册、颁发登录证书（Ecerts），颁发交易证书（Tcerts），管理证书生命周期如创建、续期与撤销与生成私钥信息等；

Fabric客户端：主要用于发起交易提案，此外我们可以把Fabric客户端理解成一个用户，如果用户具有管理员的权限，它还负责网络配置与节点管理，包括初始化与更新配置、启动和停止节点等，同时，还负责通道管理（创建、更新、查询等）与链码生命周期管理（安装、实例化、调用、升级等）。

目前，Fabric客户端包括CLI（Command Line Interface）命令行客户端和多种语言SDK客户端，如Node.js、Go、Java、Python等，负责与其他服务节点进行交互，提供配置操作、通道操作、链码操作、节点操作、日志操作等相关API接口，支持开发丰富的应用程序。

### 2.3 架构介绍

从**应用层**视角来看，Hyperledger Fabric为开发人员提供了CLI命令行终端、事件模块、客户端SDK、链码API等接口，为上层应用提供了身份管理、账本管理、交易管理、智能合约管理等区块链服务，如下图所示，



Hyperledger Fabric逻辑架构示意图

身份管理：获取用户注册证书及其私钥，用于身份验证、消息签名与验签等；

账本管理：提供多种方式查询与保存账本数据，如查询指定区块号的区块数据；

交易管理：构造并发送签名提案消息请求背书，检查合法后请求交易排序，并打包成区块，验证交易后提交账本；

智能合约管理：基于链码API编写智能合约程序，安装链码并实例化（部署）后，通过调用链码请求执行更改状态的操作。

从服务层视角看，Hyperledger Fabric提供了成员关系服务、共识服务、链码服务、安全与密码服务等服务，具体如下：

成员关系服务：Fabric-CA节点提供成员登录注册服务，接收申请并授权新用户证书与私钥等，对身份证书生命周期进行管理。MSP组件基于身份证书实现对成员等资源实体进行认证等权限管理操作，同一个MSP组件对象内的成员拥有共同信任的根证书（其中上述3个核心逻辑中，Membership服务用来管理节点身份、隐私、confidentiality 和 auditability。在一个 non-permissioned的区块链网络里，参与者不要求授权，所有的节点被视作一样，都可以提交一个transaction，去把这些交易存到区块（blocks）中。Membership Service将一个 non-permissioned的区块链网络变成一个permissioned的区块链网络，凭借着Public Key Infrastructure (PKI)、去中心和一致性）；

区块链服务：通过Endorser背书节点模拟执行提案消息，请求对模拟执行结果等签名进行背书，再提交到Orderer节点共识组件对交易进行排序并打包出块，然后交由Committer记账节点验证交易并提交账本。同时，基于Gossip消息协议提供P2P网络通信机制，实现高效数据分发与状态同步，确保节点账本的一致性；

链码服务：基于Docker容器提供隔离运行环境执行链码，支持多种语言开发的链码程序（智能合约），具有良好的可扩展性，同时，提供完善的镜像文件仓库管理机制，支持快速环境部署与测试；

安全与密码服务：将安全与密码服务封装为BCCSP组件，提供生成密钥、消息签名与验签、加密与解密、获取哈希函数等服务功能，具有可插拔组件特性，能够扩展定制的密码安全服务算法（如国密等）。

## 3．Fabric完整交易处理流程介绍

介绍一个完整交易的数据流图。

完整交易流程本质上就是一个达成共识的过程， fabric共识达成需要三个步骤：

1. 客户端发起提案，每个peer节点模拟执行进行背书。
2. Order节点进行排序。
3. Order节点验证后生成区块交给peer节点同步账本数据。

三个步骤保证了区块链数据的一致性和正确性。

Transaction流程一：



背书节点实际上是一些特殊的peer，因为不是每个节点都会参与背书，可以根据背书策略指定。

Transaction流程二：



在智能合约实例化的时候我们指定了背书策略，每个peer节点模拟执行的结果会返回给客户端用户，用户在搜集到节点模拟执行的读写集(Read and Write set)后，根据背书策略来决定是否为合法请求。

Transaction流程三：

用户提交背书后的读写集（RW Sets）给order节点，在同一时间可能会有不同用户提交背书后的读写集，这个提交操作是并行执行的。



排序服务再介绍：区块链需要解决双花问题（double spending）,实际上把并行的事情变成线性，把可能引起不一致的并行操作串行化。已买火车票为例，同一张票同一个座位可能会被两个不同的待售点卖出，解决的思路有很多种，卖票前先打电话询问其他的代售点确认不冲突才可以卖，这是同步锁的方式。或者约定了第一家 售票点只能在8点-9点出票，第二家待售点只能在9点-10点间卖票，这是通过令牌的方式解决。另一种方式是所有出票操作交给一个中心的机构进行出票，中心出票之前会检查是否还有票，没有票就会出票失败。Hyper ledger Fabric 采用的就是中心机构出票的方式，所以它效率很高，没有挖矿的概念。

排序机制：

Solo模式，单一的order节点排序组件，不能容错，只能用于测试环境下；

Kafka模式， 阿帕奇开源流式消息处理服务平台，提供非拜占庭容错，可容忍部分节点宕机，由于搭建比较复杂，在fabric 2.0 之后被弃用。

SBFT简单拜占庭容错，容忍集群中order节点有不超过1/3的错误，实际中用到的比较少。

RAFT只能容纳故障节点(CFT)，不容纳作恶节点。

Transaction流程四：

Ordering节点要验证读写集，排序生成区块，最终把区块交给 所有peer节点，让它们更新账本数据。



Transaction流程五：

Committing peer 验证读写集跟当前世界状态是否一致，一致的话就会更新账本数据，世界状态也会发生变化，如果不一致，不会更新账本，但交易会被记录，世界状态不发生变化。



Transaction流程六：

最后，Committing peer 会异步地通知用户，交易是成功还是失败。监听这个回调，就可以知道数据有没有被写入成功。

