**Anchor Peer / 锚节点**

所有其他peer可以发现并与之通信的channel上的一个peer。channel上的每个Member都有一个（或者对个，以防止单点故障）Anchor Peer，允许属于不同Member的peer发现channel上的现有其他peer。

**Block / 区块**

在一个channel上与前一个block加密连接的一组有序交易集。

**Chain / 链**

chain就是block之间以hash连接为结构的交易日志。peer从order service接收交易block，并根据背书策略和并发冲突标记block上的交易是否有效，然后将该block追加到peer文件系统中的hash chain上。

**Chaincode / 链码**

Chaincode是一个运行在ledger上的软件，对资产和交易指令（业务逻辑）编码来修改资产。

**Channel / 通道**

channel是在Fabric网络中的私有区块链区域，实现了数据的隔离和保密。channel特定的ledger在该channel上的所有peer之间共享，交易方必须有对该channel正确的认证才能与该ledger交互。channel由Configuration Block定义。

**Commitment / 提交**

一个channel中的每个peer都会验证交易的有序block，然后将block提交（写或追加）至该channel上ledger的各个副本。peer还会标记每个block中的每个交易是否有效。

**Concurrency Control Version Check / CCVC**

CCVC是一种channel中各peer间保持状态同步的方法。peer并行执行交易，在提交至ledger之前会检查在执行期间读到的数据有没有被修改。如果读取的数据在执行和提交之间被改变，就会引发CCVC冲突，该交易就会被标记为无效，值不会更新到状态数据库中。

**Configuration Block / 配置区块**

包含 system chain (ordering service) 或 channel 的Member和策略配置数据。对某个channel或整个网络的任何修改（比如，Member离开或加入）都将导致一个新的Configuration Block追加到适当的chain中。这个Configuration Block会包含 genesis block的内容加上增量。

**Consensus / 共识**

Consensus是贯穿整个交易流程的广义术语，其用于产生对于order的共识和确认构成block的所有交易的正确性。

**Current State / 当前状态**

ledger的current state表示其chain交易log中所有key的最新值。peer会将处理过的block中的每个交易对应的修改value提交到ledger的current state，由于current state表示channel所知的所有最新的k-v，所以current state也被称为World State。Chaincode执行交易proposal就是针对的current state。

**Dynamic Membership / 动态成员**

Fabric支持动态添加/移除members、peers和ordering服务节点，而不会影响整个网络的操作性。当业务关系调整或因各种原因需添加/移除实体时，Dynamic Membership至关重要。

**Endorsement / 背书**

Endorsement 是指一个peer执行一个交易并返回YES/NO给生成交易proposal的client app 的过程。chaincode具有相应的endorsement policies，其中指定了endorsing peer。

**Endorsement policy / 背书策略**

Endorsement policy定义了依赖于特定chaincode执行交易的channel上的peer和响应结果（endorsements）的必要组合条件（即返回Yes或No的条件）。Endorsement policy可指定对于某一chaincode，可以对交易背书的最小背书节点数或者最小背书节点百分比。背书策略由背书节点基于应用程序和对抵御不良行为的期望水平来组织管理。在install和instantiate Chaincode（deploy tx）时需要指定背书策略。

**Fabric-ca**

Fabric-ca是默认的证书管理组件，它向网络成员及其用户颁发基于PKI的证书。CA为每个成员颁发一个根证书（rootCert），为每个授权用户颁发一个注册证书（eCert），为每个注册证书颁发大量交易证书（tCerts）。

**Genesis Block / 初始区块**

Genesis Block是初始化区块链网络或channel的配置区块，也是链上的第一个区块。

**Gossip Protocol / Gossip协议**

Gossip数据传输协议有三项功能：1）管理peer发现和channel成员；2）channel上的所有peer间广播账本数据；3）channel上的所有peer间同步账本数据。

**Initialize / 初始化**

一个初始化chaincode程序的方法。

**Install / 安装**

将chaincode放到peer的文件系统的过程。*（译注：即将ChaincodeDeploymentSpec信息存到chaincodeInstallPath/chaincodeName.chainVersion文件中）*

**Instantiate / 实例化**

启动chaincode容器的过程。*（译注：在lccc中将ChaincodeData保存到state中，然后deploy Chaincode并执行Init方法）*

**Invoke / 调用**

用于调用chaincode内的函数。Chaincode invoke就是一个交易proposal，然后执行模块化的流程（背书、共识、 验证、 提交）。invoke的结构就是一个函数和一个参数数组。

**Leading Peer / 主导节点**

每一个Member在其订阅的channel上可以拥有多个peer，其中一个peer会作为channel的leading peer代表该Member与ordering service通信。ordering service将block传递给leading peer，该peer再将此block分发给同一member下的其他peer。

**Ledger / 账本**

Ledger是个channel的chain和由channel中每个peer维护的world state。*（这个解释有点怪）*

**Member / 成员**

拥有网络唯一根证书的合法独立实体。像peer节点和app client这样的网络组件会链接到一个Member。

**Membership Service Provider / MSP**

MSP是指为client和peer提供证书的系统抽象组件。Client用证书来认证他们的交易；peer用证书认证其交易背书。该接口与系统的交易处理组件密切相关，旨在使已定义的成员身份服务组件以这种方式顺利插入而不会修改系统的交易处理组件的核心。

**Membership Services / 成员服务**

成员服务在许可的区块链网络上认证、授权和管理身份。在peer和order中运行的成员服务的代码都会认证和授权区块链操作。它是基于PKI的MSP实现。

fabric-ca组件实现了成员服务，来管理身份。特别的，它处理ECert和TCert的颁发和撤销。ECert是长期的身份凭证；TCert是短期的身份凭证，是匿名和不可链接的。

**Ordering Service / 排序服务或共识服务**

将交易排序放入block的节点的集合。ordering service独立于peer流程之外，并以先到先得的方式为网络上所有的channel作交易排序。ordering service支持可插拔实现，目前默认实现了SOLO和Kafka。ordering service是整个网络的公用binding，包含与每个Member相关的加密材料。

**Peer / 节点**

一个网络实体，维护ledger并运行Chaincode容器来对ledger执行read/write操作。peer由Member拥有和维护。

**Policy / 策略**

有背书策略，校验策略，区块提交策略，Chaincode管理策略和网络/通道管理策略。

**Proposal / 提案**

一种针对channel中某peer的背书请求。每个proposal要么是Chaincode instantiate要么是Chaincode invoke。

**Query / 查询**

对于current state中某个key的value的查询请求。

**Software Development Kit / SDK**

SDK为开发人员提供了一个结构化的库环境，用于编写和测试链码应用程序。SDK完全可以通过标准接口实现配置和扩展，像签名的加密算法、日志框架和state存储这样的组件都可以轻松地实现替换。SDK API使用gRPC进行交易处理，成员服务、节点遍历以及事件处理都是据此与fabric通信。目前SDK支持Node.js、Java和Python。

**State Database / stateDB**

为了从Chaincode中高效的读写，Current state 数据存储在stateDB中，包括levelDB和couchDB。

**System Chain / 系统链**

包含在系统级定义网络的配置区块。系统链存在于ordering service中，与channel类似，具有包含以下信息的初始配置：MSP信息、策略和信息配置。对整个网络的任何变化（例如新的Org加入或者添加新的Ordering节点）将导致新的配置区块被添加到系统链。

系统链可看做是一个channel或一组channel的公用binding。例如，金融机构的集合可以形成一个财团（以system chain表示），然后根据其相同或不同的业务创建channel。

**Transaction / 交易**

Chaincode的invoke或instantiate操作。Invoke是从ledger中请求read/write set；Instantiate是请求在peer上启动Chaincode容器。