# Hyperledger Fabric

# Key Concepts
# 关键概念



## 一、Introduction

## 一、介绍

Hyperledger Fabric is a platform for distributed ledger solutions underpinned by a modular architecture delivering high degrees of confidentiality, resiliency, flexibility, and scalability. It is designed to support pluggable implementations of different components and accommodate the complexity and intricacies that exist across the economic ecosystem.

Hyperledger Fabric 是一个分布式账本解决方案平台，由模块化架构支撑，提供高度机密性、弹性、灵活性和可扩展性。它旨在支持不同组件的可插拔实现，并适应经济生态系统中存在的复杂性和复杂性。

We recommend first-time users begin by going through the rest of the introduction below in order to gain familiarity with how blockchains work and with the specific features and components of Hyperledger Fabric.

为了熟悉区块链的工作方式以及 Hyperledger Fabric 的特定功能和组件，我们建议首次使用的用户从以下介绍的其余部分开始。

Once comfortable — or if you're already familiar with blockchain and Hyperledger Fabric — go to Getting Started and from there explore the demos, technical specifications, APIs, etc

一旦了解了——或者如果你已经熟悉了区块链和超级账本结构——那就开始吧，从那里开始探索演示、技术规范、API 等。

## 1、What is a Blockchain?

## 1、什么是区块链？

**A Distributed Ledger**

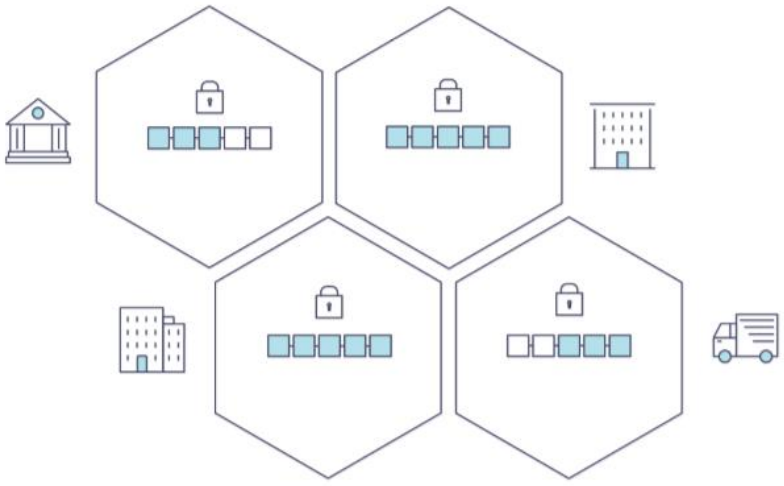
**分布式账本**

At the heart of a blockchain network is a distributed ledger that records all the transactions that take place on the network.

区块链网络的核心是一个分布式账本，记录网络上发生的所有交易。

A blockchain ledger is often described as decentralized because it is replicated across many network participants, each of whom collaborate in its maintenance. We'll see that decentralization and collaboration are powerful attributes that mirror the way businesses exchange goods and services in the real world.

区块链分类账通常被称为分散式分类账，因为它在许多网络参与者之间复制，每个参与者都在维护中协作。我们将看到，权力下放和协作是反映企业在现实世界中交换商品和服务方式的强大属性。



In addition to being decentralized and collaborative, the information

recorded to a blockchain is append-only, using cryptographic techniques that guarantee that once a transaction has been added to the ledger it cannot be modified. This property of “immutability” makes it simple to determine the provenance of information because participants can be sure information has not been changed after the fact. It’s why blockchains are sometimes described as systems of proof.

除了分散和协作之外，记录到区块链的信息只能附加，使用加密技术确保一旦交易被添加到分类账中，它就不能被修改。“不可变”的这一属性使得确定信息来源变得简单，因为参与者可以确保信息在事实发生后没有被更改。这就是为什么区块链有时被描述为证明系统的原因。

**Smart Contracts**

**智能合约**

To support the consistent update of information — and to enable a whole host of ledger functions (transacting, querying, etc) — a blockchain network uses smart contracts to provide controlled access to the ledger.

为了支持信息的一致更新——并实现整个分类账功能（交易、查询等）——区块链网络使用智能合约提供对分类账的受控访问。


Smart contracts are not only a key mechanism for encapsulating information

and keeping it simple across the network, they can also be written to allow participants to execute certain aspects of transactions automatically.

智能合约不仅是封装信息并使其在整个网络中保持简单的关键机制，还可以编写智能合约以允许参与者自动执行事务的某些方面。

A smart contract can, for example, be written to stipulate the cost of shipping an item where the shipping charge changes depending on how quickly the item arrives. With the terms agreed to by both parties and written to the ledger, the appropriate funds change hands automatically when the item is received.

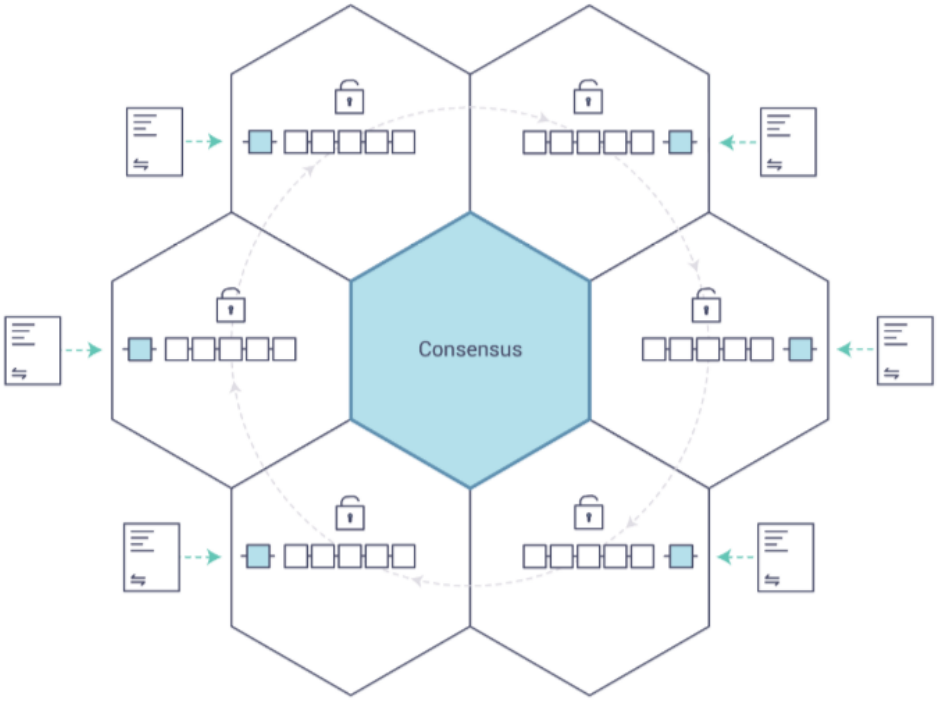
例如，可以编写智能合同，以规定装运项目的成本，其中装运费用根据项目到达的速度变化。在双方同意并写入分类账的条件下，收到项目时相应的资金自动换手。

**Consensus**

**共识**

The process of keeping the ledger transactions synchronized across the network — to ensure that ledgers update only when transactions are approved by the appropriate participants, and that when ledgers do update, they update with the same transactions in the same order — is called consensus.

在整个网络中保持分类账交易同步的过程——确保分类账仅在交易被适当的参与者批准时更新，并且当分类账更新时，它们以相同的顺序用相同的交易更新——被称为共识。




You'll learn a lot more about ledgers, smart contracts and consensus later. For now, it's enough to think of a blockchain as a shared, replicated transaction system which is updated via smart contracts and kept consistently synchronized through a collaborative process called consensus.

稍后您将了解更多关于分类账、智能合约和共识的信息。现在，把区块链看作一个共享的、复制的交易系统已经足够了，它通过智能合约进行更新，并通过一个称为共识的协作过

程保持一致的同步。

## 2、Why is a Blockchain useful?

## 2、为什么区块链有用

**Today’s Systems of Record**

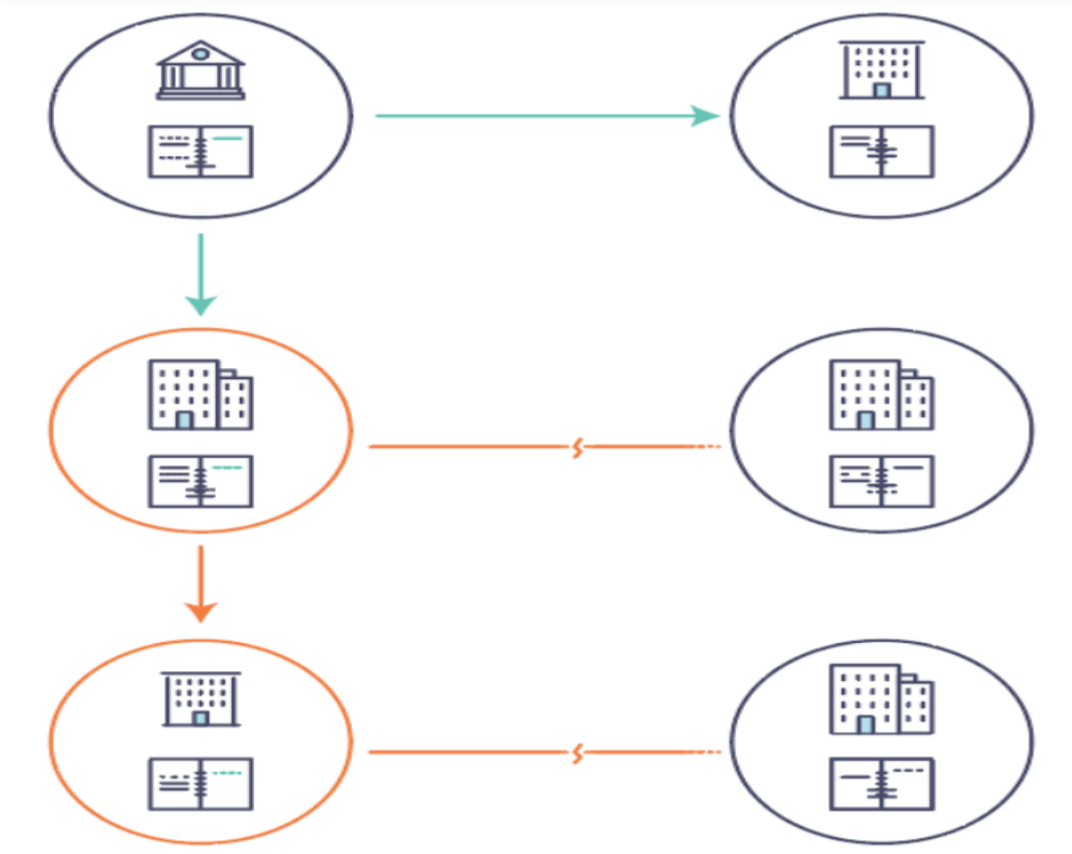
**今天的记录系统**

The transactional networks of today are little more than slightly updated versions of networks that have existed since business records have been kept. The members of a business network transact with each other, but they maintain separate records of their transactions. And the things they’re transacting — whether it’s Flemish tapestries in the 16th century or the securities of today — must have their provenance established each time they’re sold to ensure that the business selling an item possesses a chain of title verifying their ownership of it.

今天的交易性网络只不过是自保留业务记录以来存在的网络的稍微更新版本而已。商业网络中的成员彼此进行交易，但他们保留各自的交易记录。他们交易的东西——无论是 16 世纪的佛兰芒挂毯还是今天的证券——必须在每次出售时都确定其出处，以确保出售物品的企业拥有一系列所有权，以验证其所有权。

What you’re left with is a business network that looks like this:

剩下的就是这样一个商业网络：



Modern technology has taken this process from stone tablets and paper folders to hard drives and cloud platforms, but the underlying structure is the same.

Unified systems for managing the identity of network participants do not exist, establishing provenance is so laborious it takes days to clear securities transactions (the world volume of which is numbered in the many trillions of dollars), contracts must be signed and executed manually, and every database in the system contains unique information and therefore represents a single point of failure.

现代技术已经将这一过程从石版和纸质文件夹转变为硬盘和云平台，但底层结构是相同的。管理网络参与者身份的统一系统不存在，建立出处非常困难，证券交易的清算需要数天时间（全球交易量以万亿美元计），合同必须手工签署和执行，系统中的每个数据库都包含唯一的信息。因此代表了一个单一的故障点。

It’s impossible with today’s fractured approach to information and process sharing to build a system of record that spans a business network, even though the needs of visibility and trust are clear.

当今信息和流程共享的分散方法不可能构建跨越业务网络的记录系统，即使可见性和信任的需求是明确的。

**The Blockchain Difference**

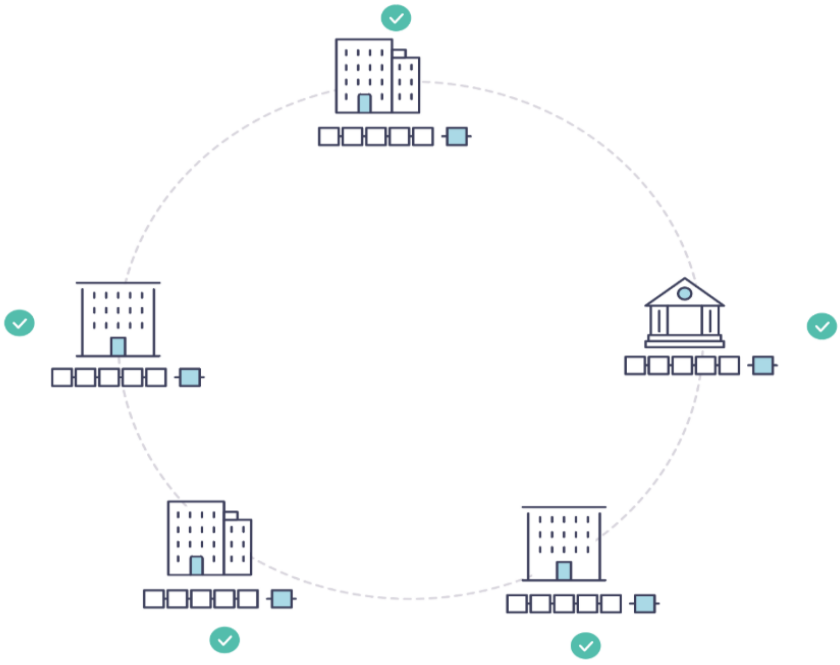
**区块链的不同**

What if, instead of the rat’s nest of inefficiencies represented by the “modern” system of transactions, business networks had standard methods for establishing identity on the network, executing transactions, and storing data? What if establishing the provenance of an asset could be determined by looking through a list of transactions that, once written, cannot be changed, and can therefore be trusted?

如果商业网络没有“现代”交易系统所代表的低效率，而是有了在网络上建立身份、执行交易和存储数据的标准方法，该怎么办？如果可以通过查看一个交易列表来确定资产的出处，该列表一旦被写入，就不能更改，因此可以被信任，那该怎么办？

That business network would look more like this:

这个商业网络看起来更像这样：



This is a blockchain network, wherein every participant has their own replicated copy of the ledger. In addition to ledger information being shared, the processes which update the ledger are also shared. Unlike today's systems, where a participant's private programs are used to update their private ledgers, a blockchain system has shared programs to update shared ledgers.

这是一个区块链网络，每个参与者都有自己的分类账副本。除了共享分类帐信息外，还共享更新分类帐的过程。不同于今天的系统，参与者的私有程序被用来更新他们的私有分类账，区块链系统有共享程序来更新共享分类账。

With the ability to coordinate their business network through a shared ledger, blockchain networks can reduce the time, cost, and risk associated with private information and processing while improving trust and visibility.

区块链网络具有通过共享分类账协调其业务网络的能力，可以减少与私人信息和处理相关的时间、成本和风险，同时提高信任度和可视性。

You now know what blockchain is and why it's useful. There are a lot of other details that are important, but they all relate to these fundamental ideas of the sharing of information and processes.

你现在知道什么是区块链，为什么它有用。还有许多其他重要的细节，但它们都与信息和过程共享的基本思想有关。

## 3、What is Hyperledger Fabric?

## 3、什么是Hyperledger Fabric

The Linux Foundation founded the Hyperledger project in 2015 to advance cross-industry blockchain technologies. Rather than declaring a single blockchain standard, it encourages a collaborative approach to developing blockchain technologies via a community process, with intellectual property rights that encourage open development and the adoption of key standards over time.

Linux基金会在2015成立了HyroundGrand项目，以推动跨行业的BBC技术。与宣布单一区块链标准不同，它鼓励通过社区流程开发区块链技术的协作方法，知识产权鼓励开放式开发，并随着时间的推移采用关键标准。

Hyperledger Fabric is one of the blockchain projects within Hyperledger. Like other blockchain technologies, it has a ledger, uses smart contracts, and is a system by which participants manage their transactions.

Hyperledger Fabric是Hyperledger中的区块链项目之一。与其他区块链技术一样，它有一个分类账，使用智能合约，是一个参与者管理其交易的系统。

Where Hyperledger Fabric breaks from some other blockchain systems is that it is private and permissioned. Rather than an open permissionless system that allows unknown identities to participate in the network (requiring protocols like "proof of work" to validate transactions and secure the network), the members of a Hyperledger Fabric network enroll through a trusted Membership Service Provider (MSP).

Hyperledger Fabric与其他一些区块链系统的区别在于它是私有的，并且是被许可的。Hyperledger Fabric网络的成员通过受信任的成员服务提供商（MSP）注册，而不是允许未知身份参与网络的开放式无权限系统（需要"工作证明"等协议来验证交易和保护网络）。

Hyperledger Fabric also offers several pluggable options. Ledger data can be stored in multiple formats, consensus mechanisms can be swapped in and out, and different MSPs are supported.

Hyperledger 结构还提供了几种可插拔选项。分类账数据可以多种格式存储，共识机制可以交换进出，支持不同的 MSP。

Hyperledger Fabric also offers the ability to create **channels**, allowing a group of participants to create a separate ledger of transactions. This is an especially important option for networks where some participants might be competitors and not want every transaction they make — a special price they're offering to some participants and not others, for example — known to every participant. If two participants form a channel, then those participants — and no others — have copies of the ledger for that channel.

Hyperledger 结构还提供了创建渠道的能力，允许一组参与者创建单独的交易分类账。对于网络来说，这是一个特别重要的选择，其中一些参与者可能是竞争对手，而不是他们所做的每一笔交易——例如，他们提供给某些参与者而不是其他参与者的一个特别价格——每个参与者都知道。如果两个参与者组成一个渠道，那么这些参与者——而不是其他人——就拥有该渠道的分类账副本。

**Shared Ledger**

**共享分类账**

Hyperledger Fabric has a ledger subsystem comprising two components: the world state and the transaction log. Each participant has a copy of the ledger to every Hyperledger Fabric network they belong to.

HyperledgeFabric 有一个由两个组件组成的分类账子系统：世界状态和交易日志。每个参与者都有一份他们所属的每个超级账本结构网络的账本副本。

The world state component describes the state of the ledger at a given point in time. It's the database of the ledger. The transaction log component records all transactions which have resulted in the current value of the world state; it's the update history for the world state. The ledger, then, is a combination of the world state database and the transaction log history.

世界状态组件描述了在给定时间点的分类帐状态。这是分类帐的数据库。交易日志组件记录了导致当前世界状态值的所有交易；它是世界状态的更新历史记录。然后，分类帐是世界状态数据库和交易日志历史的组合。

The ledger has a replaceable data store for the world state. By default, this is a LevelDB key-value store database. The transaction log does not need to be pluggable. It simply records the before and after values of the ledger database being used by the blockchain network.

分类帐有一个可替换的世界状态数据存储。默认情况下，这是一个 LEVELDB 键值存储数据库。交易日志不需要是可插入的。它只记录区块链网络使用的分类账数据库的前后值。

**Smart Contracts**

**智能合约**

Hyperledger Fabric smart contracts are written in **chaincode** and are invoked by an application external to the blockchain when that application needs to interact with the ledger. In most cases, chaincode interacts only with the database component of the ledger, the world state (querying it, for

example), and not the transaction log.

超账本结构智能合约以链码形式编写，当应用程序需要与账本交互时，由区块链外部的应用程序调用。在大多数情况下，chaincode 只与分类账的数据库组件、世界状态（例如查询它）交互，而不与交易日志交互。

Chaincode can be implemented in several programming languages. Currently, Go and Node are supported.

链码可以用几种编程语言实现。目前支持 go 和 node。

**Privacy**

**隐私**

Depending on the needs of a network, participants in a Business-to-Business (B2B) network might be extremely sensitive about how much information they share. For other networks, privacy will not be a top concern.

根据网络的需要，企业对企业(B2B)网络的参与者可能对他们共享的信息量非常敏感。对于其他网络来说，隐私并不是首要问题。

Hyperledger Fabric supports networks where privacy (using channels) is a key operational requirement as well as networks that are comparatively open.

Hyperledger 结构支持隐私(使用通道)是关键操作要求的网络，以及相对开放的网络。

**Consensus**

**共识**

Transactions must be written to the ledger in the order in which they occur, even though they might be between different sets of participants within the network. For this to happen, the order of transactions must be established and a method for rejecting bad transactions that have been inserted into the ledger in error (or maliciously) must be put into place.

交易必须按照它们发生的顺序写入分类帐，即使它们可能在网络中的不同参与者组之间。要做到这一点，必须建立交易顺序，并制定拒绝错误（或恶意）插入分类账的不良交易的方法。

This is a thoroughly researched area of computer science, and there are many ways to achieve it, each with different trade-offs. For example, PBFT (Practical Byzantine Fault Tolerance) can provide a mechanism for file replicas to communicate with each other to keep each copy consistent, even in the event of corruption. Alternatively, in Bitcoin, ordering happens through a process called mining where competing computers race to solve a cryptographic puzzle which defines the order that all processes subsequently build upon.

这是一个深入研究的计算机科学领域，有很多方法可以实现它，每个方法都有不同的权衡。例如，pbft（实际的拜占庭容错）可以提供一种机制，使文件副本能够相互通信，以保持每个副本的一致性，即使在发生损坏的情况下也是如此。或者，在比特币中，排序是通过一个称为挖掘的过程进行的，在这个过程中，竞争计算机竞相解决一个密码难题，该难题定义了所有进程随后建立的顺序。

Hyperledger Fabric has been designed to allow network starters to choose a consensus mechanism that best represents the relationships that exist between participants. As with privacy, there is a spectrum of needs; from networks that are highly structured in their relationships to those that are more peer-to-peer.

Hyperledger Fabric 的设计允许网络初学者选择最能代表参与者之间存在关系的共识

机制。与隐私一样，有一系列的需求；从关系中高度结构化的网络到更为对等的网络。

We’ll learn more about the Hyperledger Fabric consensus mechanisms, which currently include SOLO and Kafka.

我们将进一步了解 Hyperledger Fabric 共识机制，目前包括 Solo 和 Kafka。

## 4、Where can I learn more?

## 4、我在哪里可以学到更多？

- Identity (conceptual documentation)

A conceptual doc that will take you through the critical role identities play in a Fabric network (using an established PKI structure and x.509 certificates).

•身份（概念文档）

一个概念文档，它将带您了解结构网络中的关键角色标识（使用已建立的 PKI 结构和 X.509 证书）。

- Membership (conceptual documentation)

Talks through the role of a Membership Service Provider (MSP), which converts identities into roles in a Fabric network.

•会员资格（概念文档）

通过成员服务提供者（MSP）的角色进行讨论，MSP 将身份转换为 Fabric 网络中的角色。

- Peers (conceptual documentation)

Peers — owned by organizations — host the ledger and smart contracts and make up the physical structure of a Fabric network.

•节点（概念文档）

由组织拥有的对等方托管分类帐和智能合约，并组成 Fabric 网络的物理结构。

- Building Your First Network (tutorial)

Learn how to download Fabric binaries and bootstrap your own sample network with a sample script. Then tear down the network and learn how it was constructed one step at a time.

•构建第一个网络（教程）

了解如何下载 Fabric 二进制文件并使用示例脚本引导您自己的示例网络。然后拆掉网络，一步一步地学习它是如何构建的。

- Writing Your First Application (tutorial)

Deploys a very simple network — even simpler than Build Your First Network — to use with a simple smart contract and application.

•编写第一个应用程序（教程）

部署一个非常简单的网络-甚至比构建您的第一个网络更简单-与一个简单的智能合约和应用程序一起使用。

- Transaction Flow

A high level look at a sample transaction flow.

•交易流程

高级查看示例交易流。

- Hyperledger Fabric Model

A high level look at some of components and concepts brought up in this introduction as well as a few others and describes how they work together in a sample transaction flow.

•Hyperledger Fabric 模型

从高层次上看一下本介绍中提出的一些组件和概念，以及其他一些组件和概念，并描述它们如何在示例交易流中一起工作

# 二、Hyperledger Fabric Functionalities

# 二、Hyperledger Fabric 功能

Hyperledger Fabric is an implementation of distributed ledger technology (DLT) that delivers enterprise-ready network security, scalability, confidentiality and performance, in a modular blockchain architecture. Hyperledger Fabric delivers the following blockchain network functionalities:

Hyperledger Fabric 是分布式账本技术（DLT）的一种实现，它在模块化的区块链体系结构中提供企业就绪的网络安全性、可扩展性、机密性和性能。Hyperledger Fabric 提供以下区块链网络功能：

## 1、Identity management

## 1、身份管理

To enable permissioned networks, Hyperledger Fabric provides a membership identity service that manages user IDs and authenticates all participants on the network. Access control lists can be used to provide additional layers of permission through authorization of specific network operations. For example, a specific user ID could be permitted to invoke a chaincode application, but be blocked from deploying new chaincode.

为了启用授权网络，Hyperledger Fabric 提供了一个成员身份服务，用于管理用户 ID 并对网络上的所有参与者进行身份验证。访问控制列表可用于通过特定网络操作的授权提供额外的权限层。例如，可以允许特定的用户 ID 调用链码应用程序，但禁止部署新的链码。

## 2、Privacy and confidentiality

## 2、隐私和保密

Hyperledger Fabric enables competing business interests, and any groups that require private, confidential transactions, to coexist on the same permissioned network. Private channels are restricted messaging paths that can be used to provide transaction privacy and confidentiality for specific subsets of network members. All data, including transaction, member and channel information, on a channel are invisible and inaccessible to any network members not explicitly granted access to that channel.

Hyperledger Fabric 使竞争性的商业利益，以及任何需要私人、机密交易的集团能够共存于同一授权网络上。专用通道是受限制的消息传递路径，可用于为特定的网络成员子集提供事务隐私和机密性。通道上的所有数据，包括事务、成员和通道信息，对于未明确授予该通道访问权的任何网络成员都是不可见的和不可访问的。

## 3、Efficient processing

## 3. 高效处理

Hyperledger Fabric assigns network roles by node type. To provide concurrency

and parallelism to the network, transaction execution is separated from transaction ordering and commitment. Executing transactions prior to ordering them enables each peer node to process multiple transactions simultaneously. This concurrent execution increases processing efficiency on each peer and accelerates delivery of transactions to the ordering service.

Hyperledger Fabric 按节点类型分配网络角色。为了向网络提供并发性和并行性，交易执行与交易排序和承诺分离。在排序之前执行交易可以使每个对等节点同时处理多个事务。这种并发执行提高了每个对等端的处理效率，并加速了向订购服务交付事务。

In addition to enabling parallel processing, the division of labor unburdens ordering nodes from the demands of transaction execution and ledger maintenance, while peer nodes are freed from ordering (consensus) workloads. This bifurcation of roles also limits the processing required for authorization and authentication; all peer nodes do not have to trust all ordering nodes, and vice versa, so processes on one can run independently of verification by the other.

除了支持并行处理之外，分工还从交易执行和分类帐维护的需求中消除了排序节点的负担，而对等节点则从排序（一致）工作负载中解放出来。角色的这种分叉也限制了授权和身份验证所需的处理；所有对等节点不必信任所有排序节点，反之亦然，因此一个节点上的进程可以独立于另一个节点的验证运行。

## 4、Chaincode functionality

## 4、智能合约功能

Chaincode applications encode logic that is invoked by specific types of transactions on the channel. Chaincode that defines parameters for a change of asset ownership, for example, ensures that all transactions that transfer ownership are subject to the same rules and requirements. System chaincode is distinguished as chaincode that defines operating parameters for the entire channel. Lifecycle and configuration system chaincode defines the rules for the channel; endorsement and validation system chaincode defines the requirements for endorsing and validating transactions.

链式代码应用程序对通道上特定类型的交易调用的逻辑进行编码。例如，定义资产所有权更改参数的链码可确保所有转移所有权的交易都遵循相同的规则和要求。系统链码与定义整个通道操作参数的链码不同。生命周期和配置系统链码定义了通道的规则；认可和验证系统链码定义了认可和验证事务的要求。

## 5、Modular design

## 5、模块化设计

Hyperledger Fabric implements a modular architecture to provide functional choice to network designers. Specific algorithms for identity, ordering (consensus) and encryption, for example, can be plugged in to any Hyperledger Fabric network. The result is a universal blockchain architecture that any industry or public domain can adopt, with the assurance that its networks will be interoperable across market, regulatory and geographic boundaries.

Hyperledger Fabric 实现了模块化架构，为网络设计师提供功能选择。例如，身份、排序（共识）和加密的特定算法可以插入到任何超级账本结构网络中。其结果是任何行业或公共领域都可以采用通用的区块链体系结构，并保证其网络在市场、监管和地理边界之间具有互操作性。

# 三、Hyperledger Fabric Model

# 三、Hyperledger Fabric 模型

This section outlines the key design features woven into Hyperledger Fabric that fulfill its promise of a comprehensive, yet customizable, enterprise blockchain solution:

本节概述了 Hyperledger Fabric 中的关键设计功能，这些功能实现了其全面、可定制的企业区块链解决方案的承诺：

- **Assets** — Asset definitions enable the exchange of almost anything with monetary value over the network, from whole foods to antique cars to currency futures.
- **Chaincode** — Chaincode execution is partitioned from transaction ordering, limiting the required levels of trust and verification across node types, and optimizing network scalability and performance.
- **Ledger Features** — The immutable, shared ledger encodes the entire transaction history for each channel, and includes SQL-like query capability for efficient auditing and dispute resolution.
- **Privacy** — Channels and private data collections enable private and confidential multi-lateral transactions that are usually required by competing businesses and regulated industries that exchange assets on a common network.
- **Security & Membership Services** — Permissioned membership provides a trusted blockchain network, where participants know that all transactions can be detected and traced by authorized regulators and auditors.
- **Consensus** — A unique approach to consensus enables the flexibility and scalability needed for the enterprise.
- 资产-资产定义可以通过网络交换几乎所有具有货币价值的东西，从全食品到古董车再到货币期货。
- chaincode—chaincode 的执行与交易顺序是分开的，限制了跨节点类型所需的信任和验证级别，并优化了网络的可伸缩性和性能。
- 分类帐功能-不可变的共享分类帐对每个通道的整个交易历史进行编码，并包括类似 SQL 的查询功能，以有效地审计和解决争议。

- 隐私-渠道和私人数据收集能够实现私人和机密的多边交易，这通常是竞争企业和在公共网络上交换资产的受监管行业所要求的。
- 安全和会员服务-许可会员提供了一个可信的区块链网络，参与者知道所有交易都可以由授权的监管者和审计员检测和跟踪。
- 共识-达成共识的独特方法能够实现企业所需的灵活性和可扩展性。

## 1、Assets

## 1、资产

Assets can range from the tangible (real estate and hardware) to the intangible (contracts and intellectual property). Hyperledger Fabric provides the ability to modify assets using chaincode transactions.

资产范围从有形（不动产和硬件）到无形（合同和知识产权）。Hyperledger 结构提供了使用链代码事务修改资产的能力。

Assets are represented in Hyperledger Fabric as a collection of key-value pairs, with state changes recorded as transactions on a Channel ledger. Assets can be represented in binary and/or JSON form.

资产在 Hyperledger Fabric 中表示为关键值对的集合，状态更改记录为通道分类账上的交易。资产可以用二进制和/或 JSON 形式表示。

You can easily define and use assets in your Hyperledger Fabric applications using the Hyperledger Composer tool.

您可以使用 HyperledgeComposer 工具轻松定义和使用 HyperledgeFabric 应用程序中的资产。

## 2、Chaincode

## 2、智能合约

Chaincode is software defining an asset or assets, and the transaction instructions for modifying the asset(s); in other words, it’s the business logic. Chaincode enforces the rules for reading or altering key-value pairs or other state database information. Chaincode functions execute against the ledger’s current state database and are initiated through a transaction proposal. Chaincode execution results in a set of key-value writes (write set) that can be submitted to the network and applied to the ledger on all peers.

chaincode 是定义一个或多个资产的软件，以及修改资产的交易指令；换句话说，它是业务逻辑。chaincode 强制执行读取或更改键值对或其他状态数据库信息的规则。chaincode 函数针对分类账的当前状态数据库执行，并通过交易建议启动。链码执行会产生一组键值写入（写入集），可以提交到网络并应用到所有对等方的分类帐。

3、Ledger Features

3、分类账特征

The ledger is the sequenced, tamper-resistant record of all state transitions in the fabric. State transitions are a result of chaincode invocations (‘transactions’) submitted by participating parties. Each transaction results

in a set of asset key-value pairs that are committed to the ledger as creates, updates, or deletes.

分类帐是结构中所有状态转换的顺序、防篡改记录。状态转换是参与方提交的链码调用（“交易”）的结果。每个交易都会产生一组资产键值对，这些资产键值对在创建、更新或删除时提交到分类帐。

The ledger is comprised of a blockchain ( ‘chain’ ) to store the immutable, sequenced record in blocks, as well as a state database to maintain current fabric state. There is one ledger per channel. Each peer maintains a copy of the ledger for each channel of which they are a member.

分类帐由一个区块链（“链”）组成，以块形式存储不可变的、排序的记录，以及一个状态数据库，以维护当前结构状态。每个频道有一个分类帐。每个对等方都为其所属的每个渠道维护一份分类账副本。

Some features of a Fabric ledger:

Fabric 分类账的一些特点：

- Query and update ledger using key-based lookups, range queries, and composite key queries
- Read-only queries using a rich query language (if using CouchDB as state database)
- Read-only history queries — Query ledger history for a key, enabling data provenance scenarios
- Transactions consist of the versions of keys/values that were read in chaincode (read set) and keys/values that were written in chaincode (write set)
- Transactions contain signatures of every endorsing peer and are submitted to ordering service
- Transactions are ordered into blocks and are “delivered” from an ordering service to peers on a channel
- Peers validate transactions against endorsement policies and enforce the policies
- Prior to appending a block, a versioning check is performed to ensure that states for assets that were read have not changed since chaincode execution time
- There is immutability once a transaction is validated and committed
- A channel’s ledger contains a configuration block defining policies, access control lists, and other pertinent information
- Channels contain Membership Service Provider instances allowing for crypto materials to be derived from different certificate authorities
- 使用基于键的查找、范围查询和组合键查询查询和更新分类帐
- 使用丰富的查询语言的只读查询（如果使用 CouchDB 作为状态数据库）

- 只读历史记录查询-查询键的分类帐历史记录，启用数据来源方案
- 交易由在链码（读取集）中读取的键/值版本和在链码（写入集）中写入的键/值版本组成。
- 交易包含每个认可对等方的签名并提交给订购服务
- 交易按块排序，并从订购服务“传递”到通道上的对等方。
- 对等方根据认可策略验证交易并强制执行策略
- 在附加块之前，将执行版本控制检查，以确保在执行 chaincode 之后读取的资产的状态没有更改。
- 一旦一个事务被验证和提交，就不可变了。
- 通道的分类帐包含一个配置块，定义策略、访问控制列表和其他相关信息。
- 通道包含允许从不同证书颁发机构派生加密材料的成员身份服务提供程序实例

See the Ledger topic for a deeper dive on the databases, storage structure, and “query-ability.”

有关数据库、存储结构和“查询能力”的详细信息，请参阅分类账主题。

## 3、Privacy

## 3、隐私

Hyperledger Fabric employs an immutable ledger on a per-channel basis, as well as chaincode that can manipulate and modify the current state of assets (i.e. update key-value pairs). A ledger exists in the scope of a channel — it can be shared across the entire network (assuming every participant is operating on one common channel) — or it can be privatized to include only a specific set of participants.

hyperledger 结构使用了一个不可变的分类账（基于每个通道），以及可以操作和修改资产当前状态（即更新键值对）的链代码。分类帐存在于一个渠道的范围内-它可以在整个网络中共享（假设每个参与者都在一个公共渠道上操作）-或者它可以私有化，只包括一组特定的参与者。

In the latter scenario, these participants would create a separate channel and thereby isolate/segregate their transactions and ledger. In order to solve scenarios that want to bridge the gap between total transparency and privacy, chaincode can be installed only on peers that need to access the asset states to perform reads and writes (in other words, if a chaincode is not installed on a peer, it will not be able to properly interface with the ledger).

在后一种情况下，这些参与者将创建一个单独的渠道，从而隔离/分离他们的交易和分类账。为了解决希望弥合整体透明度和隐私之间的鸿沟的方案，只能在需要访问资产状态以执行读写操作的对等机上安装链码（换句话说，如果对等机上未安装链码，它将无法与分类帐正确接口）。

When a subset of organizations on that channel need to keep their transaction data confidential, a private data collection (collection) is used to segregate this data in a private database, logically separate from the channel ledger, accessible only to the authorized subset of organizations.

当该渠道上的组织的子集需要对其交易数据保密时，将使用私有数据收集（收集）将该数据隔离在私有数据库中，该数据库在逻辑上与渠道分类账分开，仅对授权的组织子集可访问。

Thus, channels keep transactions private from the broader network whereas collections keep data private between subsets of organizations on the channel.

因此，渠道使来自更广泛网络的交易保持私有，而收集使渠道上组织的子集之间的数据保持私有。

To further obfuscate the data, values within chaincode can be encrypted (in part or in total) using common cryptographic algorithms such as AES before sending transactions to the ordering service and appending blocks to the ledger. Once encrypted data has been written to the ledger, it can be decrypted only by a user in possession of the corresponding key that was used to generate the cipher text. For further details on chaincode encryption, see the Chaincode for Developers topic.

为了进一步混淆数据，可以使用常见的加密算法（如 AES）对链码中的值进行加密（部分或全部),然后将交易发送到订购服务并将块追加到分类帐。一旦加密数据被写入分类帐，它只能由拥有用于生成密码文本的相应密钥的用户解密。有关链码加密的更多详细信息，请参阅 ChaincodeforDevelopers 主题。

See the Private Data topic for more details on how to achieve privacy on your blockchain network.

有关如何在区块链网络上实现隐私的更多详细信息，请参阅私有数据主题。

## 4、Security & Membership Services

## 4、安全和会员服务

Hyperledger Fabric underpins a transactional network where all participants have known identities. Public Key Infrastructure is used to generate cryptographic certificates which are tied to organizations, network components, and end users or client applications. As a result, data access control can be manipulated and governed on the broader network and on channel levels. This “permissioned” notion of Hyperledger Fabric, coupled with the existence and capabilities of channels, helps address scenarios where privacy and confidentiality are paramount concerns.

Hyperledger Fabric 支持所有参与者都具有已知身份的交易网络。公钥基础结构用于生成与组织、网络组件和最终用户或客户端应用程序绑定的加密证书。因此，数据访问控制可以在更广泛的网络和通道级别上进行操作和管理。这种“许可”的超级账本结构概念，加上渠道的存在和功能，有助于解决隐私和保密性是首要问题的场景。

See the Membership Service Providers (MSP) topic to better understand cryptographic implementations, and the sign, verify, authenticate approach used in Hyperledger Fabric.

请参阅会员服务提供商(MSP)主题以更好地了解加密实现，以及在 Hyperledger Fabric 中使用的签名、验证和身份验证方法。

## 5、Consensus

## 5、共识

In distributed ledger technology, consensus has recently become synonymous with a specific algorithm, within a single function. However, consensus encompasses more than simply agreeing upon the order of transactions, and this differentiation is highlighted in Hyperledger Fabric through its fundamental role in the entire transaction flow, from proposal and endorsement, to ordering, validation and commitment. In a nutshell, consensus is defined as the full-circle verification of the correctness of a set of transactions comprising a block.

在分布式账本技术中，共识最近已成为单个函数内特定算法的同义词。然而，共识不仅仅是简单地就交易顺序达成一致，而且这种差异在 Hyperledger Fabric 中得到强调，通过其在整个交易流程中的基本角色，从建议和认可，到订购、验证和承诺。简言之，共识被定义为对包含一个块的一组交易的正确性进行全循环验证。

Consensus is achieved ultimately when the order and results of a block’s transactions have met the explicit policy criteria checks. These checks and balances take place during the lifecycle of a transaction, and include the usage of endorsement policies to dictate which specific members must endorse a certain transaction class, as well as system chaincodes to ensure that these policies are enforced and upheld. Prior to commitment, the peers will employ these system chaincodes to make sure that enough endorsements are present, and that they were derived from the appropriate entities. Moreover, a versioning check will take place during which the current state of the ledger is agreed or consented upon, before any blocks containing transactions are appended to the ledger. This final check provides protection against double spend operations and other threats that might compromise data integrity, and allows for functions to be executed against non-static variables.

当区块交易的顺序和结果满足明确的政策标准检查时，最终达成共识。这些检查和平衡发生在交易的生命周期中，包括使用认可政策来规定哪些特定成员必须认可某个交易类别，以及系统链码以确保这些政策得到执行和支持。在承诺之前，同行将使用这些系统链码，以确保存在足够的认可，并且这些认可来自适当的实体。此外，在将包含交易的任何块附加到分类帐之前，将进行版本控制检查，在此期间对分类帐的当前状态进行商定或同意。此最终检查提供了对可能损害数据完整性的重复开销操作和其他威胁的保护，并允许对非静态变量执行函数。

In addition to the multitude of endorsement, validity and versioning checks that take place, there are also ongoing identity verifications happening in all directions of the transaction flow. Access control lists are implemented on hierarchical layers of the network (ordering service down to channels), and payloads are repeatedly signed, verified and authenticated as a transaction proposal passes through the different architectural components. To conclude, consensus is not merely limited to the agreed upon order of a batch of transactions; rather, it is an overarching characterization that is achieved as a byproduct of the ongoing verifications that take place during a transaction’s journey from proposal to commitment.

除了进行大量的认可、有效性和版本控制检查外，还在事务流的各个方向进行身份验证。

访问控制列表是在网络的层次层（将服务向下订购到通道）上实现的，当事务提案通过不同的体系结构组件时，有效负载会被重复地签名、验证和验证。综上所述，共识不仅限于一批交易的商定顺序;更重要的是,它是在交易从提议到承诺的过程中进行的持续验证的副产品。

Check out the Transaction Flow diagram for a visual representation of consensus.

查看事务流程图以直观地表示一致意见。

# 四、Blockchain network

# 四、区块链网络

This topic will describe, at a conceptual level, how Hyperledger Fabric allows organizations to collaborate in the formation of blockchain networks. If you’re an architect, administrator or developer, you can use this topic to get a solid understanding of the major structure and process components in a Hyperledger Fabric blockchain network. This topic will use a manageable worked example that introduces all of the major components in a blockchain network. After understanding this example you can read more detailed information about these components elsewhere in the documentation, or try building a sample network.

本主题将在概念层面描述超账本结构如何允许组织在区块链网络的形成过程中进行协作。如果您是架构师、管理员或开发人员，您可以使用此主题深入了解超级账本结构区块链网络中的主要结构和流程组件。本主题将使用一个可管理的工作示例，介绍区块链网络中的所有主要组件。在理解了这个例子之后，您可以在文档的其他地方阅读关于这些组件的更详细的信息，或者尝试构建一个示例网络。

After reading this topic and understanding the concept of policies, you will have a solid understanding of the decisions that organizations need to make to establish the policies that control a deployed Hyperledger Fabric network. You’ll also understand how organizations manage network evolution using declarative policies - a key feature of Hyperledger Fabric. In a nutshell, you’ll understand the major technical components of Hyperledger Fabric and the decisions organizations need to make about them.

阅读本主题并理解政策的概念后，您将对组织建立控制已部署的 Hyperledger Fabric 网络的政策所需做出的决策有一个扎实的理解。您还将了解组织如何使用声明性策略（hyperledger 结构的一个关键特性）管理网络演进。简言之，您将了解 Hyperledger 结构的主要技术组件以及组织需要做出的决策。

## 1、What is a blockchain network?

## 1、什么是区块链网路

A blockchain network is a technical infrastructure that provides ledger and smart contract (chaincode) services to applications. Primarily, smart contracts are used to generate transactions which are subsequently distributed to every peer node in the network where they are immutably recorded on their copy of the ledger. The users of applications might be end users using client applications or blockchain network administrators.

区块链网络是为应用程序提供分类账和智能合约(链码)服务的技术基础设施。主要是，智能合约用于生成交易，随后将交易分发到网络中的每个对等节点，在该节点上，这些交易将永久记录在其分类账副本上。应用程序的用户可能是使用客户端应用程序或区块链网络管理员的最终用户。

In most cases, multiple organizations come together as a consortium to form the network and their permissions are determined by a set of policies that are agreed by the consortium when the network is originally configured. Moreover, network policies can change over time subject to the agreement of the organizations in the consortium, as we' ll discover when we discuss the concept of modification policy.

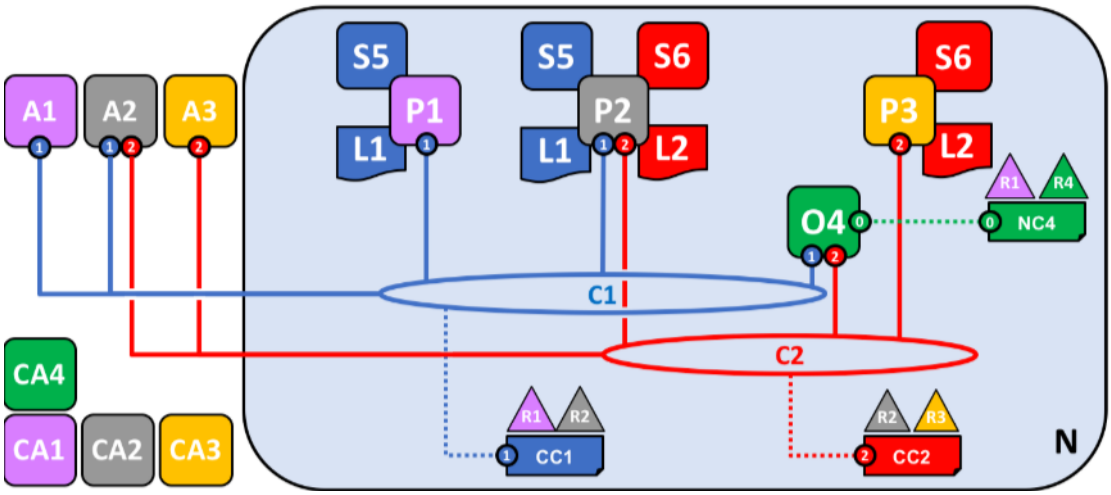
在大多数情况下，多个组织作为一个联合体聚集在一起形成网络，它们的权限由一组策略决定，这些策略在最初配置网络时由联合体商定。此外，网络策略可以随着时间的变化而变化，这取决于联合体中组织的同意，正如我们在讨论修改策略的概念时所发现的那样。

## 2、The sample network

## 2、样本网络

Before we start, let' s show you what we' re aiming at! Here' s a diagram representing the final state of our sample network.

在我们开始之前，让我们展示一下我们的目标!这是一个表示示例网络最终状态的图表。




Don' t worry that this might look complicated! As we go through this topic, we will build up the network piece by piece, so that you see how the organizations R1, R2, R3 and R4 contribute infrastructure to the network to help form it. This infrastructure implements the blockchain network, and it is governed by policies agreed by the organizations who form the network - for example, who can add new organizations. You' ll discover how applications consume the ledger and smart contract services provided by the blockchain network.

别担心这看起来很复杂！在我们讨论这个主题时，我们将逐个构建网络，以便您了解组织 R1、R2、R3 和 R4 如何为网络提供基础设施以帮助形成网络。该基础设施实现了区块链网络，并受组成该网络的组织（例如，谁可以添加新组织）同意的政策控制。您将发现应用程序如何使用区块链网络提供的分类账和智能合约服务。

Four organizations, R1, R2, R3 and R4 have jointly decided, and written into

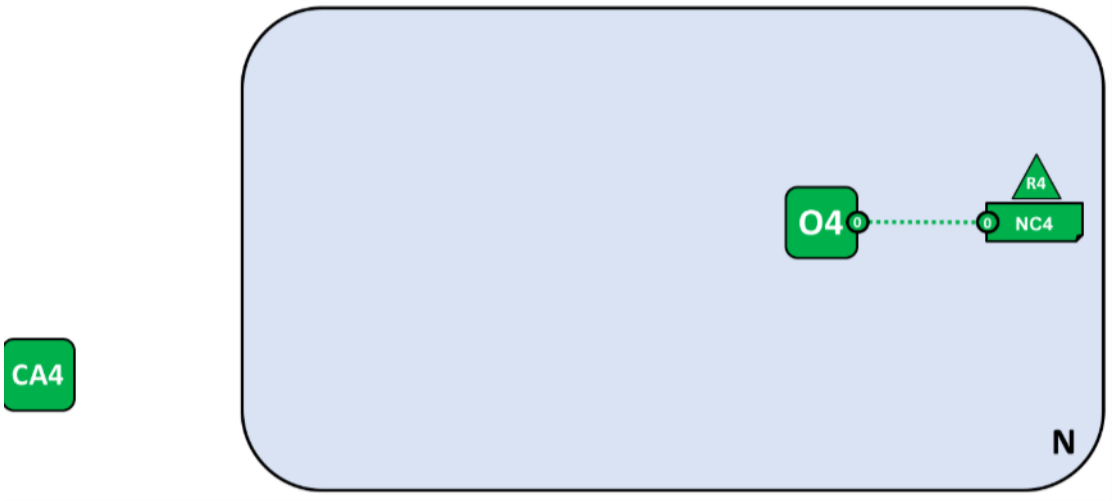
an agreement, that they will set up and exploit a Hyperledger Fabric network. R4 has been assigned to be the network initiator – it has been given the power to set up the initial version of the network. R4 has no intention to perform business transactions on the network. R1 and R2 have a need for a private communications within the overall network, as do R2 and R3. Organization R1 has a client application that can perform business transactions within channel C1. Organization R2 has a client application that can do similar work both in channel C1 and C2. Organization R3 has a client application that can do this on channel C2. Peer node P1 maintains a copy of the ledger L1 associated with C1. Peer node P2 maintains a copy of the ledger L1 associated with C1 and a copy of ledger L2 associated with C2. Peer node P3 maintains a copy of the ledger L2 associated with C2. The network is governed according to policy rules specified in network configuration NC4, the network is under the control of organizations R1 and R4. Channel C1 is governed according to the policy rules specified in channel configuration CC1; the channel is under the control of organizations R1 and R2. Channel C2 is governed according to the policy rules specified in channel configuration CC2; the channel is under the control of organizations R2 and R3. There is an ordering service O4 that services as a network administration point for N, and uses the system channel. The ordering service also supports application channels C1 and C2, for the purposes of transaction ordering into blocks for distribution. Each of the four organizations has a preferred Certificate Authority.

四个组织，即 R1、R2、R3 和 R4，共同决定，并签订协议，他们将建立和利用一个超级账本结构网络。R4 已被指定为网络发起人，它有权设置网络的初始版本。R4 无意在网络上执行业务交易。R1 和 R2 需要在整个网络中进行私有通信，R2 和 R3 也是如此。组织 R1 有一个客户机应用程序，可以在通道 C1 内执行交易业务。组织 r2 有一个客户端应用程序，可以在通道 c1 和 c2 中执行类似的工作。组织 r3 有一个客户端应用程序可以在通道 c2 上执行此操作。对等节点 p1 维护与 c1 关联的分类帐 l1 的副本。对等节点 p2 维护与 c1 关联的分类帐 l1 副本和与 c2 关联的分类帐 l2 副本。对等节点 p3 维护与 c2 关联的分类帐 l2 的副本。网络按照网络配置 NC4 中规定的策略规则进行管理，网络由组织 R1 和 R4 控制。通道 C1 按照通道配置 CC1 中指定的策略规则进行管理；通道由组织 R1 和 R2 控制。信道 C2 按照信道配置 CC2 中指定的策略规则进行管理；信道由组织 R2 和 R3 控制。有一个订购服务 O4，它作为 N 的网络管理点提供服务，并使用系统通道。订购服务还支持应用程序通道 C1 和 C2，以便将交易订购成块进行分发。这四个组织中的每一个都有一个首选的证书颁发机构。

## 3、Creating the Network

## 3、创建网络

Let’s start at the beginning by creating the basis for the network:
让我们从创建网络基础开始：




The network is formed when an orderer is started. In our example network, N, the ordering service comprising a single node, O4, is configured according to a network configuration NC4, which gives administrative rights to organization R4. At the network level, Certificate Authority CA4 is used to dispense identities to the administrators and network nodes of the R4 organization.

网络是在订购者启动时形成的。在我们的示例网络 n 中，包含单个节点 O4 的订购服务是根据网络配置 nc4 配置的，该配置赋予组织 r4 管理权限。在网络级别，证书颁发机构 CA4 用于将身份分配给 R4 组织的管理员和网络节点。

We can see that the first thing that defines a network, N, is an ordering service, O4. It's helpful to think of the ordering service as the initial administration point for the network. As agreed beforehand, O4 is initially configured and started by an administrator in organization R4, and hosted in R4. The configuration NC4 contains the policies that describe the starting set of administrative capabilities for the network. Initially this is set to only give R4 rights over the network. This will change, as we'll see later, but for now R4 is the only member of the network.

我们可以看到，定义网络 n 的第一件事是订购服务 O4。将订购服务作为网络的初始管理点是很有帮助的。按照事先约定，O4 最初由组织 r4 中的管理员配置和启动，并托管在 r4 中。配置 NC4 包含描述网络管理功能起始集的策略。最初，这被设置为仅通过网络授予 R4 权限。这将改变，正如我们稍后将看到的，但目前，R4 是网络的唯一成员。

**Certificate Authorities**

**证书颁发机构**

You can also see a Certificate Authority, CA4, which is used to issue certificates to administrators and network nodes. CA4 plays a key role in our network because it dispenses X.509 certificates that can be used to identify components as belonging to organization R4. Certificates issued by CAs can also be used to sign transactions to indicate that an organization endorses the transaction result - a precondition of it being accepted onto the ledger. Let's examine these two aspects of a CA in a little more detail.

，用于向管理员和网络节点颁发证书。CA4 在我们的网络中扮演着关键的角色，因为它分发 X.509 证书，可用于识别属于组织 R4 的组件。由 CA 颁发的证书也可用于签署交易，以

表明组织认可交易结果——这是将其接受到分类账上的前提条件。让我们更详细地研究一下 CA 的这两个方面。

Firstly, different components of the blockchain network use certificates to identify themselves to each other as being from a particular organization. That’s why there is usually more than one CA supporting a blockchain network – different organizations often use different CAs. We’re going to use four CAs in our network; one of for each organization. Indeed, CAs are so important that Hyperledger Fabric provides you with a built-in one (called Fabric-CA) to help you get going, though in practice, organizations will choose to use their own CA.

首先，区块链网络的不同组成部分使用证书彼此识别自己来自特定组织。这就是为什么通常有多个 CA 支持区块链网络的原因——不同的组织通常使用不同的 CA。我们将在网络中使用四个 CA；每个组织一个。事实上，CA 非常重要，以至于 Hyperledger Fabric 为您提供了一个内置的（称为 Fabric CA）来帮助您前进，尽管在实践中，组织会选择使用自己的 CA。

The mapping of certificates to member organizations is achieved by via a structure called a Membership Services Provider (MSP). Network configuration NC4 uses a named MSP to identify the properties of certificates dispensed by CA4 which associate certificate holders with organization R4. NC4 can then use this MSP name in policies to grant actors from R4 particular rights over network resources. An example of such a policy is to identify the administrators in R4 who can add new member organizations to the network. We don’t show MSPs on these diagrams, as they would just clutter them up, but they are very important.

将证书映射到成员组织是通过称为成员服务提供者（MSP）的结构实现的。网络配置 NC4 使用一个命名的 MSP 来标识由 CA4 分配的证书的属性，CA4 将证书持有者与组织 R4 关联起来。然后，NC4 可以在策略中使用这个 MSP 名称来授予来自 R4 的参与者对网络资源的特定权限。这种策略的一个例子是识别 R4 中可以向网络添加新成员组织的管理员。我们不会在这些图表上显示 MSP，因为它们只会使它们变得杂乱无章，但它们非常重要。

Secondly, we’ll see later how certificates issued by CAs are at the heart of the transaction generation and validation process. Specifically, X.509 certificates are used in client application transaction proposals and smart contract transaction responses to digitally sign transactions. Subsequently the network nodes who host copies of the ledger verify that transaction signatures are valid before accepting transactions onto the ledger.

其次，我们稍后将看到由 CA 颁发的证书是如何成为交易生成和验证过程的核心的。具体来说，X.509 证书用于客户端应用程序事务建议和对数字签名事务的智能合约事务响应。随后，托管分类账副本的网络节点会在接受分类账上的交易之前验证交易签名是否有效。

Let’s recap the basic structure of our example blockchain network. There’s a resource, the network N, accessed by a set of users defined by a Certificate Authority CA4, who have a set of rights over the resources in the network N as described by policies contained inside a network configuration NC4. All of this is made real when we configure and start the ordering service node O4.

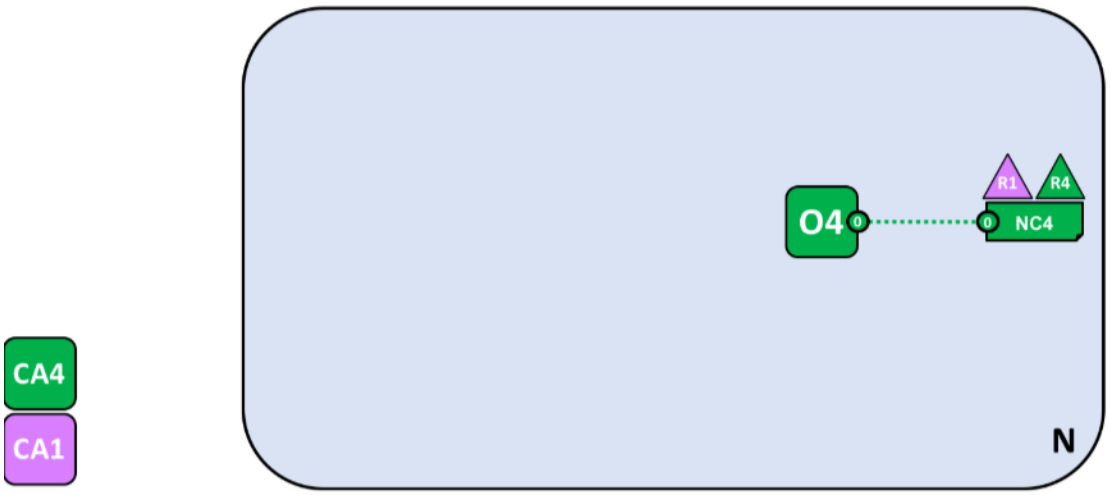
让我们回顾一下我们的示例区块链网络的基本结构。有一个资源，网络 n，由一组由证书颁发机构 CA4 定义的用户访问，这些用户对网络 n 中的资源拥有一组权限，如网络配置 NC4 中包含的策略所述。当我们配置和启动订购服务节点 O4 时，所有这些都是真实的。

## 4、Adding Network Administrators

## 4、添加网路管理员

NC4 was initially configured to only allow R4 users administrative rights over the network. In this next phase, we are going to allow organization R1 users to administer the network. Let's see how the network evolves:

NC4 最初配置为仅允许 R4 用户通过网络进行管理。在下一阶段，我们将允许组织 R1 用户管理网络。让我们看看网络是如何发展的：




Organization R4 updates the network configuration to make organization R1 an administrator too. After this point R1 and R4 have equal rights over the network configuration.

组织 r4 更新网络配置，使组织 r1 也成为管理员。在这之后，R1 和 R4 对网络配置拥有相同的权限。

We see the addition of a new organization R1 as an administrator - R1 and R4 now have equal rights over the network. We can also see that certificate authority CA1 has been added - it can be used to identify users from the R1 organization. After this point, users from both R1 and R4 can administer the network.

我们看到添加了一个新的组织 R1 作为管理员——R1 和 R4 现在在网络上拥有平等的权利。我们还可以看到添加了证书颁发机构 CA1——它可以用来识别 R1 组织中的用户。在这之后，来自 R1 和 R4 的用户都可以管理网络。

Although the orderer node, O4, is running on R4's infrastructure, R1 has shared administrative rights over it, as long as it can gain network access. It means that R1 or R4 could update the network configuration NC4 to allow the R2 organization a subset of network operations. In this way, even though R4 is running the ordering service, and R1 has full administrative rights over it, R2 has limited rights to create new consortia.

尽管订购方节点 O4 运行在 R4 的基础结构上，但 R1 对它具有共享的管理权限，只要它可以获得网络访问权。这意味着 R1 或 R4 可以更新网络配置 NC4，使 R2 组织成为网络操作的一个子集。这样，即使 R4 正在运行订购服务，并且 R1 对其具有完全的管理权限，R2 也只有有限的权限创建新的联合体。

In its simplest form, the ordering service is a single node in the network, and that's what you can see in the example. Ordering services are usually multi-node, and can be configured to have different nodes in different organizations. For example, we might run O4 in R4 and connect it to O2, a separate orderer node in organization R1. In this way, we would have a multi-site, multi-organization administration structure.

在最简单的形式中，订购服务是网络中的单个节点，这就是您在示例中看到的。订购服务通常是多节点的，可以配置为在不同的组织中具有不同的节点。例如，我们可以在 R4 中运行 O4，并将其连接到组织 R1 中单独的排序节点 O2。这样，我们就有了一个多站点、多组织的管理结构。

We'll discuss the ordering service a little more later in this topic, but for now just think of the ordering service as an administration point which provides different organizations controlled access to the network.

稍后我们将在本主题中进一步讨论订购服务，但现在仅将订购服务视为一个管理点，它提供不同组织控制的网络访问。

## 5、Defining a Consortium

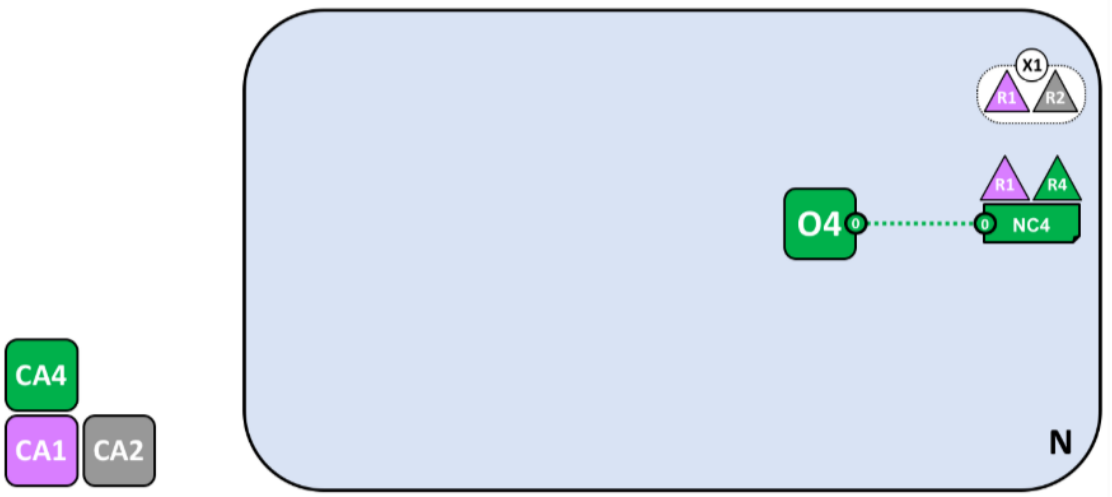
## 5、定义联合体

Although the network can now be administered by R1 and R4, there is very little that can be done. The first thing we need to do is define a consortium. This word literally means "a group with a shared destiny", so it's an appropriate choice for a set of organizations in a blockchain network.

尽管网络现在可以由 R1 和 R4 管理，但几乎没有什么可以实现的。我们需要做的第一件事就是定义一个联合体。这个词的字面意思是“有着共同命运的群体”，因此对于区块链网络中的一组组织来说，这是一个合适的选择。

Let's see how a consortium is defined:

让我们看看联合体是如何定义的：




A network administrator defines a consortium X1 that contains two members, the organizations R1 and R2. This consortium definition is stored in the network configuration NC4, and will be used at the next stage of network development. CA1 and CA2 are the respective Certificate Authorities for these organizations.

网络管理员定义一个包含两个成员的联合体 x1，即组织 r1 和 r2。这个联合体定义存储在网络配置 NC4 中，将在网络开发的下一个阶段使用。CA1 和 CA2 是这些组织各自的证书颁发机构。

Because of the way NC4 is configured, only R1 or R4 can create new consortia. This diagram shows the addition of a new consortium, X1, which defines R1 and R2 as its constituting organizations. We can also see that CA2 has been added to identify users from R2. Note that a consortium can have any number of organizational members – we have just shown two as it is the simplest configuration.

由于 NC4 的配置方式，只有 R1 或 R4 可以创建新的联合体。此图显示添加了一个新的联合体 x1，它将 r1 和 r2 定义为其组成组织。我们还可以看到，添加了 CA2 来识别来自 R2 的用户。请注意，一个联合体可以有任意数量的组织成员——我们刚刚展示了两个，因为它是最简单的配置。

Why are consortia important? We can see that a consortium defines the set of organizations in the network who share a need to transact with one another – in this case R1 and R2. It really makes sense to group organizations together if they have a common goal, and that's exactly what's happening.

为什么财团很重要？我们可以看到，一个联合体定义了网络中的一组组织，这些组织共享彼此进行交易的需求——在本例中是 R1 和 R2。如果组织有一个共同的目标，那么将它们组织在一起是很有意义的，而这正是正在发生的事情。

The network, although started by a single organization, is now controlled by a larger set of organizations. We could have started it this way, with R1, R2 and R4 having shared control, but this build up makes it easier to understand.

这个网络虽然是由一个组织发起的，但现在由一组更大的组织控制。我们本来可以这样开始的，因为 R1、R2 和 R4 共享控制权，但是这种构建使我们更容易理解。

We're now going to use consortium X1 to create a really important part of a Hyperledger Fabric blockchain – a channel.

我们现在将使用联合体 x1 来创建一个超级账本 Fabric 区块链——一个渠道的非常重要的部分。

## 6、Creating a channel for a consortium

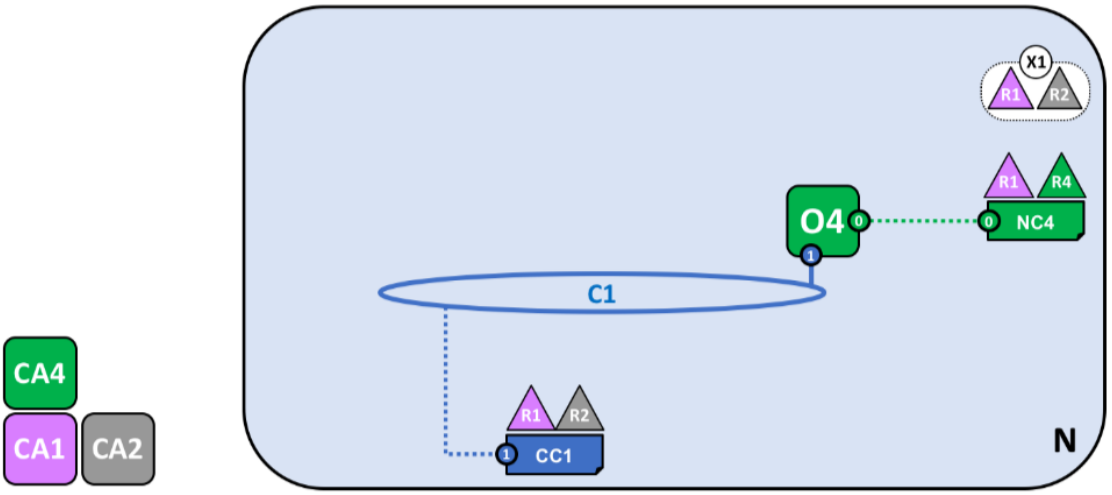
## 6、为联合体创建渠道

So let's create this key part of the Fabric blockchain network – a channel. A channel is a primary communications mechanism by which the members of a consortium can communicate with each other. There can be multiple channels in a network, but for now, we'll start with one.

因此，让我们创建这个结构区块链网络的关键部分——一个渠道。通道是一种主要的通信机制，通过它，联合体成员可以相互通信。一个网络中可以有多个频道，但现在，我们将从一个频道开始。

Let's see how the first channel has been added to the network:

让我们看看第一个频道是如何添加到网络的：




A channel C1 has been created for R1 and R2 using the consortium definition X1. The channel is governed by a channel configuration CC1, completely separate to the network configuration. CC1 is managed by R1 and R2 who have equal rights over C1. R4 has no rights in CC1 whatsoever.

已使用联合体定义 x1 为 r1 和 r2 创建了通道 c1。通道由通道配置 CC1 控制，完全独立于网络配置。CC1 由对 C1 拥有同等权利的 R1 和 R2 管理。R4 在 CC1 中没有任何权利。

The channel C1 provides a private communications mechanism for the consortium X1. We can see channel C1 has been connected to the ordering service O4 but that nothing else is attached to it. In the next stage of network development, we're going to connect components such as client applications and peer nodes. But at this point, a channel represents the potential for future connectivity.

通道 C1 为联合体 X1 提供了一个专用通信机制。我们可以看到通道 C1 已连接到订购服务 O4，但没有其他连接到它。在网络开发的下一阶段，我们将连接客户端应用程序和对等节点等组件。但在这一点上，通道代表了未来连接的潜力。

Even though channel C1 is a part of the network N, it is quite distinguishable from it. Also notice that organizations R3 and R4 are not in this channel - it is for transaction processing between R1 and R2. In the previous step, we saw how R4 could grant R1 permission to create new consortia. It's helpful to mention that R4 also allowed R1 to create channels! In this diagram, it could have been organization R1 or R4 who created a channel C1. Again, note that a channel can have any number of organizations connected to it - we've shown two as it's the simplest configuration.

尽管信道 C1 是网络 N 的一部分，但它与网络 N 是完全不同的。还请注意，组织 r3 和 r4 不在此通道中—它用于 R1 和 r2 之间的事务处理。在前面的步骤中，我们看到了 R4 如何授予 R1 创建新联合体的权限。值得一提的是，R4 还允许 R1 创建通道！在这个图中，可能是组织 R1 或 R4 创建了通道 C1。同样，请注意，一个渠道可以有任意数量的组织连接到它——我们已经展示了两个，因为它是最简单的配置。

Again, notice how channel C1 has a completely separate configuration, CC1, to the network configuration NC4. CC1 contains the policies that govern the rights that R1 and R2 have over the channel C1 - and as we've seen, R3 and R4 have no permissions in this channel. R3 and R4 can only interact with C1 if they

are added by R1 or R2 to the appropriate policy in the channel configuration CC1. An example is defining who can add a new organization to the channel. Specifically, note that R4 cannot add itself to the channel C1 – it must, and can only, be authorized by R1 or R2.

同样，请注意通道 C1 如何对网络配置 NC4 进行完全独立的配置 CC1。CC1 包含管理 R1 和 R2 对通道 C1 拥有的权限的策略—正如我们所看到的，R3 和 R4 在此通道中没有权限。r3 和 r4 只能与 c1 交互，前提是它们由 r1 或 r2 添加到通道配置 cc1 中的相应策略中。一个例子是定义谁可以向渠道添加新组织。具体来说，请注意，R4 不能将自身添加到通道 C1 中——它必须且只能由 R1 或 R2 授权。

Why are channels so important? Channels are useful because they provide a mechanism for private communications and private data between the members of a consortium. Channels provide privacy from other channels, and from the network. Hyperledger Fabric is powerful in this regard, as it allows organizations to share infrastructure and keep it private at the same time. There's no contradiction here – different consortia within the network will have a need for different information and processes to be appropriately shared, and channels provide an efficient mechanism to do this. Channels provide an efficient sharing of infrastructure while maintaining data and communications privacy.

为什么渠道如此重要？通道是有用的，因为它们为联合体成员之间的私有通信和私有数据提供了一种机制。频道提供来自其他频道和网络的隐私。在这方面，Hyperledger 结构非常强大，因为它允许组织共享基础设施，同时保持其私有性。这里没有矛盾——网络中不同的联盟需要适当地共享不同的信息和流程，而渠道提供了一种有效的机制来实现这一点。渠道在维护数据和通信隐私的同时，提供了基础设施的高效共享。

We can also see that once a channel has been created, it is in a very real sense "free from the network". It is only organizations that are explicitly specified in a channel configuration that have any control over it, from this time forward into the future. Likewise, any updates to network configuration NC4 from this time onwards will have no direct effect on channel configuration CC1; for example if consortia definition X1 is changed, it will not affect the members of channel C1. Channels are therefore useful because they allow private communications between the organizations constituting the channel. Moreover, the data in a channel is completely isolated from the rest of the network, including other channels.

我们还可以看到，一旦创建了一个频道，它就在一个非常真实的意义上"从网络中解放出来"。只有在通道配置中明确指定的组织才能控制它，从现在到将来。同样，从此时起对网络配置 nc4 的任何更新都不会对通道配置 cc1 产生直接影响；例如，如果更改了联合定义 x1，则不会影响通道 c1 的成员。因此，渠道是有用的，因为它们允许构成渠道的组织之间进行私人通信。此外，通道中的数据与网络的其他部分（包括其他通道）完全隔离。

As an aside, there is also a special system channel defined for use by the ordering service. It behaves in exactly the same way as a regular channel, which are sometimes called application channels for this reason. We don't normally need to worry about this channel, but we'll discuss a little bit more about it later in this topic.

除此之外，订购服务还定义了一个特殊的系统通道。它的行为方式与常规通道完全相同，

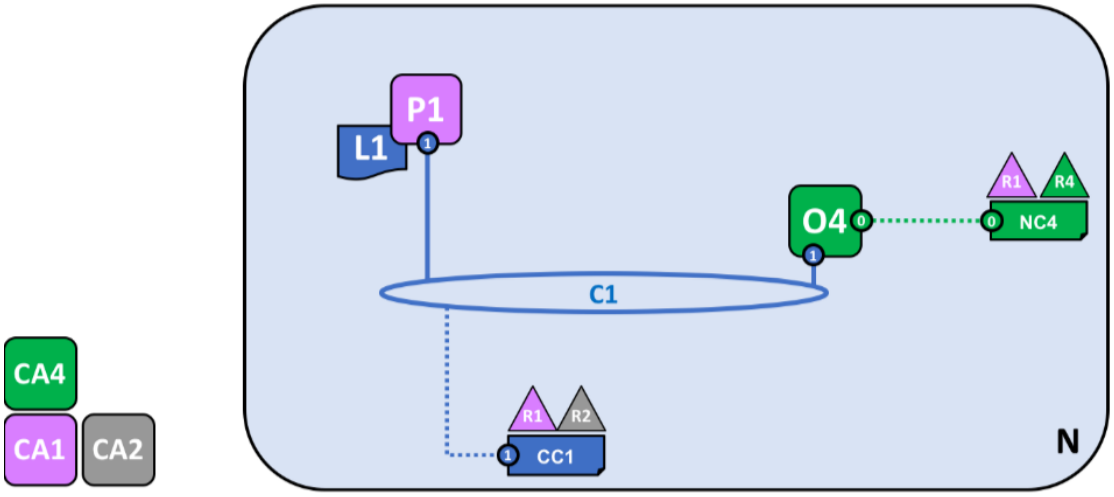
因此有时称为应用程序通道。我们通常不需要担心这个频道，但稍后我们将在本主题中对此进行更多讨论。

## 7、Peers and Ledgers

## 7、同行和分类账

Let’s now start to use the channel to connect the blockchain network and the organizational components together. In the next stage of network development, we can see that our network N has just acquired two new components, namely a peer node P1 and a ledger instance, L1.

现在让我们开始使用该渠道将区块链网络和组织组件连接在一起。在网络开发的下一个阶段，我们可以看到我们的网络 n 刚刚获得了两个新组件，即对等节点 p1 和分类帐实例 l1。




A peer node P1 has joined the channel C1. P1 physically hosts a copy of the ledger L1. P1 and O4 can communicate with each other using channel C1.

对等节点 p1 已加入通道 c1。p1 实际承载分类帐 l1 的副本。P1 和 O4 可以通过通道 C1 相互通信。

Peer nodes are the network components where copies of the blockchain ledger are hosted! At last, we’re starting to see some recognizable blockchain components! P1’s purpose in the network is purely to host a copy of the ledger L1 for others to access. We can think of L1 as being physically hosted on P1, but logically hosted on the channel C1. We’ll see this idea more clearly when we add more peers to the channel.

对等节点是承载区块链分类账副本的网络组件！最后，我们开始看到一些可识别的区块链组件！p1 在网络中的用途纯粹是为其他人提供一份分类账 l1 的副本。我们可以将 L1 看作是物理托管在 P1 上，但逻辑托管在通道 C1 上。当我们向频道添加更多的对等端时，我们会更清楚地看到这个想法。

A key part of a P1’s configuration is an X.509 identity issued by CA1 which associates P1 with organization R1. Once P1 is started, it can join channel C1 using the orderer O4. When O4 receives this join request, it uses the channel configuration CC1 to determine P1’s permissions on this channel. For example, CC1 determines whether P1 can read and/or write information to the ledger L1.

P1 配置的一个关键部分是由 CA1 颁发的 X.509 标识，它将 P1 与组织 R1 关联起来。一旦 p1 启动，它就可以使用 order o4 连接通道 c1。当 O4 收到这个连接请求时，它使用通道配置 cc1 来确定 p1 对此通道的权限。例如，cc1 确定 p1 是否可以读取和/或写入分类帐 l1 的信息。

Notice how peers are joined to channels by the organizations that own them, and though we've only added one peer, we'll see how there can be multiple peer nodes on multiple channels within the network. We'll see the different roles that peers can take on a little later.

请注意，拥有这些节点的组织如何将对等节点加入到通道中，尽管我们只添加了一个对等节点，但我们将看到网络中多个通道上如何存在多个对等节点。我们将看到同龄人稍后可以扮演的不同角色。

## 8、Applications and Smart Contract chaincode

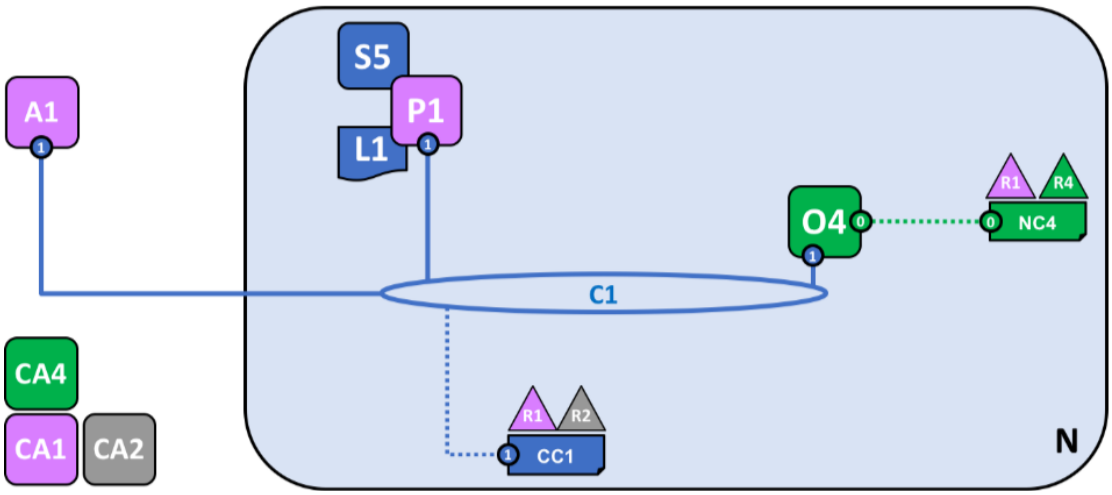
## 8、应用程序和智能合约链码

Now that the channel C1 has a ledger on it, we can start connecting client applications to consume some of the services provided by workhorse of the ledger, the peer!

既然通道 C1 上有一个分类账，那么我们就可以开始连接客户机应用程序来使用分类账的 Workhorse（对等方）提供的一些服务了！

Notice how the network has grown:

注意网络的发展：




A smart contract S5 has been installed onto P1. Client application A1 in organization R1 can use S5 to access the ledger via peer node P1. A1, P1 and O4 are all joined to channel C1, i.e. they can all make use of the communication facilities provided by that channel.

智能合约 S5 已安装到 P1 上。组织 r1 中的客户端应用程序 a1 可以使用 s5 通过对等节点 p1 访问分类帐。A1、P1 和 O4 都与信道 C1 相连，即它们都可以利用该信道提供的通信设施。

In the next stage of network development, we can see that client application A1 can use channel C1 to connect to specific network resources - in this case A1 can connect to both peer node P1 and orderer node O4. Again, see how channels

are central to the communication between network and organization components. Just like peers and orderers, a client application will have an identity that associates it with an organization. In our example, client application A1 is associated with organization R1; and although it is outside the Fabric blockchain network, it is connected to it via the channel C1.

在网络开发的下一阶段，我们可以看到客户机应用程序 A1 可以使用通道 C1 连接到特定的网络资源——在这种情况下，A1 可以连接到对等节点 P1 和订购方节点 O4。同样，请看一下通道如何成为网络和组织组件之间通信的核心。与同行和订购者一样，客户机应用程序将具有一个将其与组织关联的标识。在我们的示例中，客户机应用程序 A1 与组织 R1 关联；尽管它在结构区块链网络之外，但它通过通道 C1 连接到它。

It might now appear that A1 can access the ledger L1 directly via P1, but in fact, all access is managed via a special program called a smart contract chaincode, S5. Think of S5 as defining all the common access patterns to the ledger; S5 provides a well-defined set of ways by which the ledger L1 can be queried or updated. In short, client application A1 has to go through smart contract S5 to get to ledger L1!

现在看来，A1 可以通过 p1 直接访问分类账 l1，但实际上，所有访问都是通过一个称为智能合约链码（smart contract chaincode，s5）的特殊程序管理的。将 S5 视为定义了对分类账的所有常见访问模式；S5 提供了一组定义良好的方法，通过这些方法可以查询或更新分类账 L1。简而言之，客户端应用程序 A1 必须通过智能合约 S5 才能到达分类帐 L1！

Smart contract chaincodes can be created by application developers in each organization to implement a business process shared by the consortium members. Smart contracts are used to help generate transactions which can be subsequently distributed to the every node in the network. We’ll discuss this idea a little later; it’ll be easier to understand when the network is bigger. For now, the important thing to understand is that to get to this point two operations must have been performed on the smart contract; it must have been installed, and then instantiated.

每个组织中的应用程序开发人员可以创建智能合同链代码，以实现联合体成员共享的业务流程。智能合约用于帮助生成事务，这些事务随后可以分发到网络中的每个节点。稍后我们将讨论这个想法；当网络更大时，更容易理解。现在，要理解的重要一点是，要达到这一点，必须对智能合约执行两个操作；必须先安装智能合约，然后实例化它。

**Installing a smart contract**

**安装智能合约**

After a smart contract S5 has been developed, an administrator in organization R1 must install it onto peer node P1. This is a straightforward operation; after it has occurred, P1 has full knowledge of S5. Specifically, P1 can see the implementation logic of S5 – the program code that it uses to access the ledger L1. We contrast this to the S5 interface which merely describes the inputs and outputs of S5, without regard to its implementation.

开发智能合约 S5 后，组织 R1 中的管理员必须将其安装到对等节点 P1 上。这是一个简单的操作；在它发生之后，p1 已经完全了解了 s5。具体来说，p1 可以看到 s5 的实现逻辑——它用来访问分类账 l1 的程序代码。我们将其与仅描述 S5 输入和输出的 S5 接口进行对比，而不考虑其实现。

When an organization has multiple peers in a channel, it can choose the peers upon which it installs smart contracts; it does not need to install a smart contract on every peer.

当一个组织在一个通道中有多个对等机时，它可以选择安装智能合约的对等机；它不需要在每个对等机上安装智能合约。

**Instantiating a smart contract**

**实例化智能合约**

However, just because P1 has installed S5, the other components connected to channel C1 are unaware of it; it must first be instantiated on channel C1. In our example, which only has a single peer node P1, an administrator in organization R1 must instantiate S5 on channel C1 using P1. After instantiation, every component on channel C1 is aware of the existence of S5; and in our example it means that S5 can now be invoked by client application A1!

然而，仅仅因为 p1 安装了 s5，连接到通道 c1 的其他组件并不知道它；它必须首先在通道 c1 上实例化。在我们的示例中，只有一个对等节点 p1，组织 r1 中的管理员必须使用 p1 在通道 c1 上实例化 s5。在实例化之后，通道 C1 上的每个组件都知道存在 S5；在我们的示例中，它意味着现在可以由客户机应用程序 A1 调用 S5！

Note that although every component on the channel can now access S5, they are not able to see its program logic. This remains private to those nodes who have installed it; in our example that means P1. Conceptually this means that it's the smart contract interface that is instantiated, in contrast to the smart contract implementation that is installed. To reinforce this idea; installing a smart contract shows how we think of it being physically hosted on a peer, whereas instantiating a smart contract shows how we consider it logically hosted by the channel.

请注意，虽然通道上的每个组件现在都可以访问 S5，但它们无法看到其程序逻辑。对于安装了它的节点，这仍然是私有的；在我们的示例中，这意味着 p1。从概念上讲，这意味着它是实例化的智能合约接口，而不是安装的智能合约实现。为了强化这一想法，安装智能合约显示了我们如何看待它在物理上托管在对等端上，而实例化智能合约则显示了我们如何看待它在逻辑上托管在通道上。

**Endorsement policy**

**背书政策**

The most important piece of additional information supplied at instantiation is an endorsement policy. It describes which organizations must approve transactions before they will be accepted by other organizations onto their copy of the ledger. In our sample network, transactions can be only be accepted onto ledger L1 if R1 or R2 endorse them.

在实例化时提供的最重要的附加信息是认可策略。它描述了哪些组织必须批准交易才能被其他组织接受到其分类账副本中。在我们的样本网络中，只有在 R1 或 R2 对交易进行背书的情况下，交易才能被接受到分类账 L1 上。

The act of instantiation places the endorsement policy in channel configuration CC1; it enables it to be accessed by any member of the channel. You can read more about endorsement policies in the transaction flow topic.

实例化操作将背书策略放置在通道配置 CC1 中；它使通道的任何成员都可以访问它。您

可以在“交易流”主题中了解更多有关认可策略的信息。

Invoking a smart contract

**调用智能合约**

Once a smart contract has been installed on a peer node and instantiated on a channel it can be invoked by a client application. Client applications do this by sending transaction proposals to peers owned by the organizations specified by the smart contract endorsement policy. The transaction proposal serves as input to the smart contract, which uses it to generate an endorsed transaction response, which is returned by the peer node to the client application.

一旦智能合约安装在对等节点上并在通道上实例化，客户机应用程序就可以调用它。客户端应用程序通过向智能合约认可策略指定的组织所拥有的对等方发送交易建议来实现这一点。交易建议用作智能合约的输入，智能合约使用它生成已背书的交易响应，该响应由对等节点返回到客户端应用程序。

It’s these transactions responses that are packaged together with the transaction proposal to form a fully endorsed transaction, which can be distributed to the entire network. We’ll look at this in more detail later For now, it’s enough to understand how applications invoke smart contracts to generate endorsed transactions.

正是这些交易响应与交易建议打包在一起，形成一个完全认可的交易，可以分发到整个网络。我们稍后将对此进行更详细的研究，这足以理解应用程序如何调用智能合约来生成认可的交易。

By this stage in network development we can see that organization R1 is fully participating in the network. Its applications - starting with A1 - can access the ledger L1 via smart contract S5, to generate transactions that will be endorsed by R1, and therefore accepted onto the ledger because they conform to the endorsement policy.

在网络开发的这个阶段，我们可以看到组织 R1 完全参与了网络。其应用程序（从 A1 开始）可以通过智能合约 S5 访问分类账 L1，以生成将由 R1 背书的交易，并因此接受到分类账，因为它们符合背书政策。

## 9、Network completed

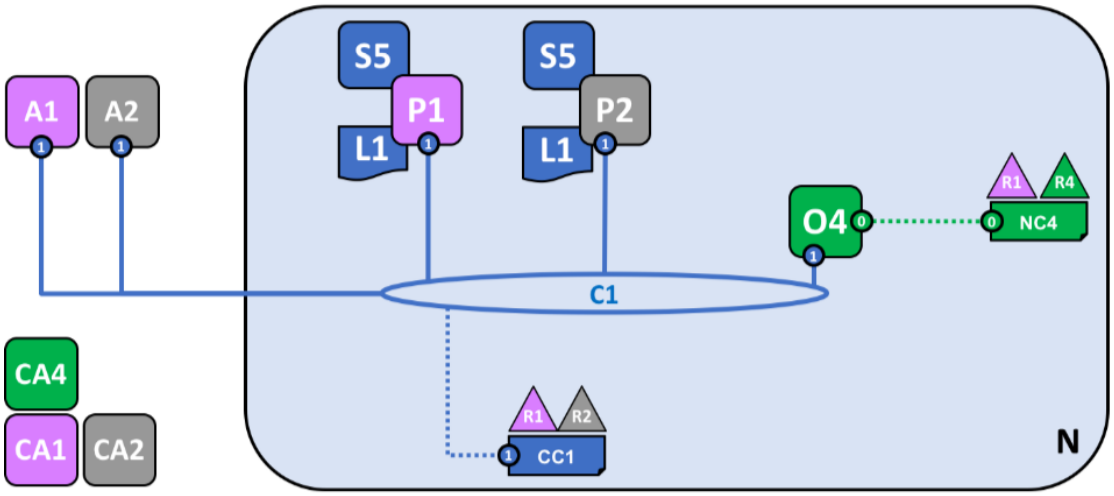
## 9、网络完成

Recall that our objective was to create a channel for consortium X1 - organizations R1 and R2. This next phase of network development sees organization R2 add its infrastructure to the network.

回想一下，我们的目标是为联盟 x1——组织 r1 和 r2 创建一个渠道。网络开发的下一个阶段将看到组织 r2 将其基础结构添加到网络中。

Let’s see how the network has evolved:

让我们看看网络是如何发展的：




The network has grown through the addition of infrastructure from organization R2. Specifically, R2 has added peer node P2, which hosts a copy of ledger L1, and chaincode S5. P2 has also joined channel C1, as has application A2. A2 and P2 are identified using certificates from CA2. All of this means that both applications A1 and A2 can invoke S5 on C1 either using peer node P1 or P2.

通过从组织 r2 中添加基础设施，网络得以增长。具体来说，R2 增加了对等节点 p2，它承载分类账 l1 的副本，以及链码 s5。P2 也加入了通道 C1，应用程序 A2 也加入了通道 C1。A2 和 P2 通过 CA2 认证。所有这些都意味着应用程序 A1 和 A2 都可以使用对等节点 P1 或 P2 在 C1 上调用 S5。

We can see that organization R2 has added a peer node, P2, on channel C1. P2 also hosts a copy of the ledger L1 and smart contract S5. We can see that R2 has also added client application A2 which can connect to the network via channel C1. To achieve this, an administrator in organization R2 has created peer node P2 and joined it to channel C1, in the same way as an administrator in R1.

我们可以看到组织 r2 在通道 c1 上添加了一个对等节点 p2。P2 还托管分类账 L1 和智能合约 S5 的副本。我们可以看到，R2 还添加了客户端应用程序 A2，它可以通过通道 C1 连接到网络。为了实现这一点，组织 r2 中的管理员创建了对等节点 p2 并将其加入通道 c1，与 R1 中的管理员相同。

We have created our first operational network! At this stage in network development, we have a channel in which organizations R1 and R2 can fully transact with each other. Specifically, this means that applications A1 and A2 can generate transactions using smart contract S5 and ledger L1 on channel C1.

我们已经创建了第一个运营网络！在网络开发的这个阶段，我们有一个渠道，在这个渠道中，组织 r1 和 r2 可以充分地相互处理。具体来说，这意味着应用程序 A1 和 A2 可以使用通道 C1 上的智能合约 S5 和分类账 L1 生成交易。

**Generating and accepting transactions**

**生成和接受交易**

In contrast to peer nodes, which always host a copy of the ledger, we see that there are two different kinds of peer nodes; those which host smart contracts and those which do not. In our network, every peer hosts a copy of the smart contract, but in larger networks, there will be many more peer nodes that do not

host a copy of the smart contract. A peer can only run a smart contract if it is installed on it, but it can know about the interface of a smart contract by being connected to a channel.

与总是承载分类账副本的对等节点不同，我们看到有两种不同类型的对等节点：承载智能合约的节点和不承载智能合约的节点。在我们的网络中，每一个对等点都承载智能合约的副本，但在更大的网络中，将有更多的对等节点不承载智能合约的副本。对等端只能在安装了智能合约的情况下运行它，但它可以通过连接到通道来了解智能合约的接口。

You should not think of peer nodes which do not have smart contracts installed as being somehow inferior. It's more the case that peer nodes with smart contracts have a special power - to help generate transactions. Note that all peer nodes can validate and subsequently accept or reject transactions onto their copy of the ledger L1. However, only peer nodes with a smart contract installed can take part in the process of transaction endorsement which is central to the generation of valid transactions.

您不应该认为没有安装智能合约的对等节点在某种程度上是劣等的。更重要的是，具有智能合约的对等节点有一种特殊的能力——帮助生成交易。请注意，所有对等节点都可以验证并随后接受或拒绝其分类账 1 副本上的交易。但是，只有安装了智能合约的对等节点才能参与交易认可过程，这是生成有效交易的核心。

We don't need to worry about the exact details of how transactions are generated, distributed and accepted in this topic - it is sufficient to understand that we have a blockchain network where organizations R1 and R2 can share information and processes as ledger-captured transactions. We'll learn a lot more about transactions, ledgers, smart contracts in other topics.

我们不需要担心在本主题中如何生成、分发和接受交易的确切细节——我们有一个区块链网络，在该网络中，组织 R1 和 R2 可以作为分类账捕获的交易共享信息和流程。我们将在其他主题中了解更多关于交易、分类账、智能合约的信息。

**Types of peers**

**节点类型**

In Hyperledger Fabric, while all peers are the same, they can assume multiple roles depending on how the network is configured. We now have enough understanding of a typical network topology to describe these roles.

在 Hyperledger Fabric 中，尽管所有对等方都是相同的，但它们可以根据网络配置的方式承担多个角色。我们现在已经对典型的网络拓扑结构有了足够的理解来描述这些角色。

- *Committing peer*. Every peer node in a channel is a committing peer. It receives blocks of generated transactions, which are subsequently validated before they are committed to the peer node's copy of the ledger as an append operation.
- *Endorsing peer*. Every peer with a smart contract *can* be an endorsing peer if it has a smart contract installed. However, to actually *be* an endorsing peer, the smart contract on the peer must be used by a client application to generate a digitally signed transaction response. The term *endorsing peer* is an explicit reference to this fact.

An endorsement policy for a smart contract identifies the organizations whose peer should digitally sign a generated transaction before it can be accepted onto a committing peer's copy of the ledger.

- 提交节点。通道中的每个对等节点都是提交对等节点。它接收生成的交易块，这些交易块随后在作为追加操作提交到对等节点的分类帐副本之前进行验证。
- 认可节点。如果安装了智能合约，则每个具有智能合约的对等方都可以是认可对等方。然而，要真正成为认可对等体，客户端应用程序必须使用对等体上的智能合约来生成数字签名事务响应。“认可对等”一词是对这一事实的明确引用。

智能合约的背书政策确定了在将生成的交易接受到提交对等方的分类账副本之前，对等方应先对其进行数字签名的组织。

These are the two major types of peer; there are two other roles a peer can adopt:

这是两种主要类型的节点；节点还可以采用两种其他角色：

- *Leader peer*. When an organization has multiple peers in a channel, a leader peer is a node which takes responsibility for distributing transactions from the orderer to the other committing peers in the organization. A peer can choose to participate in static or dynamic leadership selection.

  It is helpful, therefore to think of two sets of peers from leadership perspective - those that have static leader selection, and those with dynamic leader selection. For the static set, zero or more peers can be configured as leaders. For the dynamic set, one peer will be elected leader by the set. Moreover, in the dynamic set, if a leader peer fails, then the remaining peers will re-elect a leader.

  It means that an organization's peers can have one or more leaders connected to the ordering service. This can help to improve resilience and scalability in large networks which process high volumes of transactions.
- *Anchor peer*. If a peer needs to communicate with a peer in another organization, then it can use one of the **anchor peers** defined in the channel configuration for that organization. An organization can have zero or more anchor peers defined for it, and an anchor peer can help with many different cross-organization communication scenarios.

- 领导者节点。当一个组织在一个渠道中有多个节点时，领导者对等方是负责将交易从订购方分发到组织中其他提交对等方的节点。同伴可以选择参与静态或动态领导选择。

  因此，从领导层的角度考虑两组同事是很有帮助的——一组是静态的领导选择，另一组是动态的领导选择。对于静态集，可以将零个或多个对等机配置为前导。对于动态集，一个对等体将由该集选出领导者。此外，在动态集合中，如果一个领导者对等方失败，那么其余的对等方将重新选择领导者。

  这意味着一个组织的同行可以有一个或多个领导者连接到订购服务。这有助于提高处理大量事务的大型网络的弹性和可扩展性。
- 锚节点。如果节点需要与另一个组织中的节点通信，那么它可以使用在该组织的通道配置中定义的锚定节点之一。一个组织可以为其定义零个或多个锚定对等，锚定对等可以帮助处理许多不同的跨组织通信场景。

Note that a peer can be a committing peer, endorsing peer, leader peer and anchor peer all at the same time! Only the anchor peer is optional – for all practical purposes there will always be a leader peer and at least one endorsing peer and at least one committing peer.

请注意，节点可以同时是提交节点、认可节点、领导节点和锚定节点！只有锚定节点是可选的——出于所有实际目的，始终会有一个领导节点和至少一个认可节点和至少一个提交节点。

**Install not instantiate**

**安装未实例化**

In a similar way to organization R1, organization R2 must install smart contract S5 onto its peer node, P2. That's obvious – if applications A1 or A2 wish to use S5 on peer node P2 to generate transactions, it must first be present; installation is the mechanism by which this happens. At this point, peer node P2 has a physical copy of the smart contract and the ledger; like P1, it can both generate and accept transactions onto its copy of ledger L1.

与组织 r1 类似，组织 r2 必须将智能合约 s5 安装到其对等节点 p2 上。这是显而易见的——如果应用程序 A1 或 A2 希望在对等节点 P2 上使用 S5 来生成交易，则必须首先出现；安装是发生这种情况的机制。此时，对等节点 p2 具有智能合约和分类账的物理副本；与 p1 一样，它可以在其分类账 l1 副本上生成和接受交易。

However, in contrast to organization R1, organization R2 does not need to instantiate smart contract S5 on channel C1. That's because S5 has already been instantiated on the channel by organization R1. Instantiation only needs to happen once; any peer which subsequently joins the channel knows that smart contract S5 is available to the channel. This fact reflects the fact that ledger L1 and smart contract really exist in a physical manner on the peer nodes, and a logical manner on the channel; R2 is merely adding another physical instance of L1 and S5 to the network.

但是，与组织 R1 相比，组织 R2 不需要在通道 C1 上实例化智能合约 S5。这是因为组织

R1 已经在通道上实例化了 S5。实例化只需要发生一次；随后加入通道的任何对等方都知道该通道可以使用智能合约 S5。这一事实反映了一个事实，即分类账 L1 和智能合约确实以物理方式存在于对等节点上，并且在通道上以逻辑方式存在；R2 只是将 L1 和 S5 的另一个物理实例添加到网络中。

In our network, we can see that channel C1 connects two client applications, two peer nodes and an ordering service. Since there is only one channel, there is only one logical ledger with which these components interact. Peer nodes P1 and P2 have identical copies of ledger L1. Copies of smart contract S5 will usually be identically implemented using the same programming language, but if not, they must be semantically equivalent.

在我们的网络中，我们可以看到通道 C1 连接两个客户机应用程序、两个对等节点和一个订购服务。因为只有一个通道，所以这些组件与之交互的逻辑分类账只有一个。对等节点 p1 和 p2 具有相同的分类帐 l1 副本。智能合约 S5 的副本通常使用相同的编程语言以相同的方式实现，但如果不是，则它们必须在语义上等效。

We can see that the careful addition of peers to the network can help support increased throughput, stability, and resilience. For example, more peers in a network will allow more applications to connect to it; and multiple peers in an organization will provide extra resilience in the case of planned or unplanned outages.

我们可以看到，小心地向网络中添加对等节点有助于提高吞吐量、稳定性和恢复能力。例如，网络中更多的对等方将允许更多的应用程序连接到它；组织中的多个对等方将在计划内或计划外停机时提供额外的弹性。

It all means that it is possible to configure sophisticated topologies which support a variety of operational goals – there is no theoretical limit to how big a network can get. Moreover, the technical mechanism by which peers within an individual organization efficiently discover and communicate with each other – the gossip protocol – will accommodate a large number of peer nodes in support of such topologies.

这一切都意味着可以配置支持各种操作目标的复杂拓扑——对于一个网络可以达到多大的规模，理论上没有限制。此外，一个组织内的对等方高效发现并相互通信的技术机制——gossip 协议——将容纳大量对等节点，以支持这种拓扑结构。

The careful use of network and channel policies allow even large networks to be well-governed. Organizations are free to add peer nodes to the network so long as they conform to the policies agreed by the network. Network and channel policies create the balance between autonomy and control which characterizes a de-centralized network.

谨慎地使用网络和通道策略甚至可以很好地管理大型网络。只要符合网络同意的策略，组织就可以自由地向网络添加对等节点。网络和渠道策略在自治和控制之间建立平衡，这是非集中网络的特征。

## 10、Simplifying the visual vocabulary

## 10、 简单化的视觉词汇

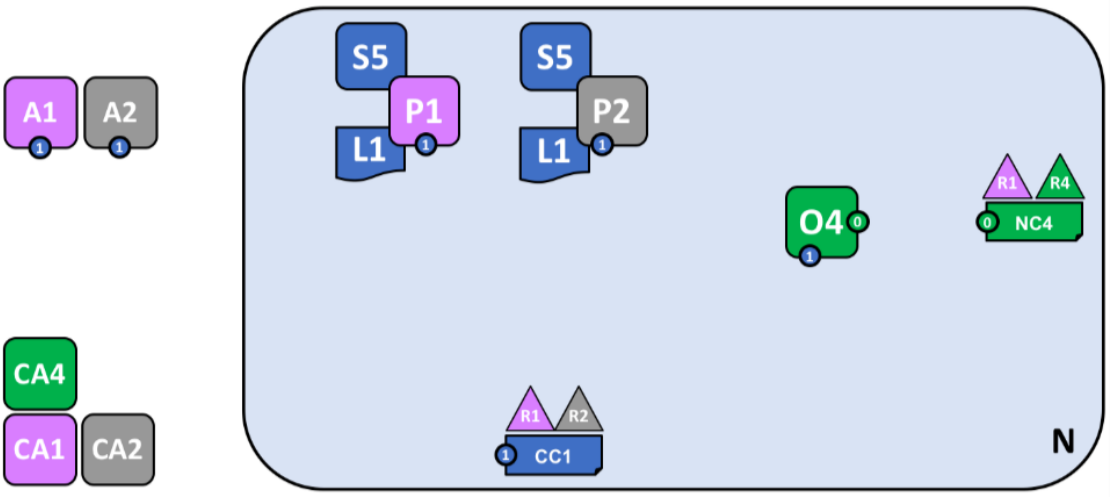
We’re now going to simplify the visual vocabulary used to represent our

sample blockchain network. As the size of the network grows, the lines initially used to help us understand channels will become cumbersome. Imagine how complicated our diagram would be if we added another peer or client application, or another channel?

我们现在将简化用于表示我们的示例区块链网络的可视化词汇表。随着网络规模的扩大，最初用来帮助我们了解频道的线路将变得繁琐。想象一下，如果我们添加另一个对等或客户机应用程序，或者另一个通道，我们的图表会有多复杂？

That's what we're going to do in a minute, so before we do, let's simplify the visual vocabulary. Here's a simplified representation of the network we've developed so far:

这就是我们马上要做的，所以在做之前，让我们简化视觉词汇表。以下是迄今为止我们开发的网络的简化表示：




The diagram shows the facts relating to channel C1 in the network N as follows: Client applications A1 and A2 can use channel C1 for communication with peers P1 and P2, and orderer O4. Peer nodes P1 and P2 can use the communication services of channel C1. Ordering service O4 can make use of the communication services of channel C1. Channel configuration CC1 applies to channel C1.

图中显示了与网络 n 中通道 c1 相关的事实：客户端应用程序 a1 和 a2 可以使用通道 c1 与对等端 p1 和 p2 以及订购方 o4 通信。对等节点 p1 和 p2 可以使用通道 c1 的通信服务。订购服务 O4 可以利用信道 C1 的通信服务。信道配置 CC1 适用于信道 C1。

Note that the network diagram has been simplified by replacing channel lines with connection points, shown as blue circles which include the channel number. No information has been lost. This representation is more scalable because it eliminates crossing lines. This allows us to more clearly represent larger networks. We've achieved this simplification by focusing on the connection points between components and a channel, rather than the channel itself.

请注意，通过用连接点替换通道线简化了网络图，如包含通道号的蓝色圆圈所示。没有丢失任何信息。这种表示更具可扩展性，因为它消除了交叉线。这使我们能够更清楚地表示更大的网络。我们通过关注组件和通道之间的连接点而不是通道本身来实现这种简化。

## 11、Adding another consortium definition

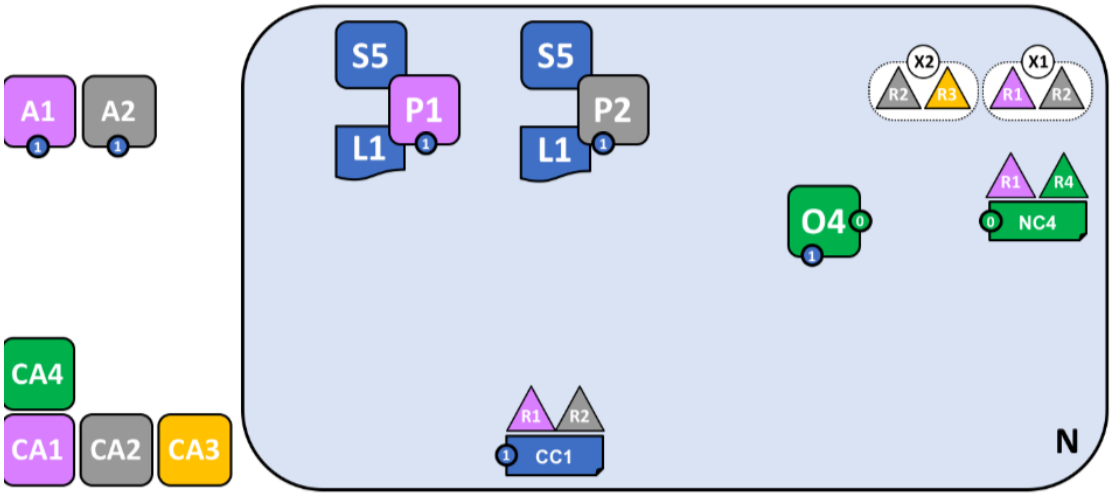
## 11、添加另一个联合体定义

In this next phase of network development, we introduce organization R3. We’re going to give organizations R2 and R3 a separate application channel which allows them to transact with each other. This application channel will be completely separate to that previously defined, so that R2 and R3 transactions can be kept private to them.

在网络开发的下一个阶段，我们将介绍组织 r3。我们将为组织 r2 和 r3 提供一个单独的应用程序通道，允许它们彼此进行交易。此应用程序通道将完全独立于先前定义的通道，以便 R2 和 R3 事务可以对它们保密。

Let’s return to the network level and define a new consortium, X2, for R2 and R3:

让我们回到网络级别，为 r2 和 r3 定义一个新的联合体 x2：




A network administrator from organization R1 or R4 has added a new consortium definition, X2, which includes organizations R2 and R3. This will be used to define a new channel for X2.

组织 R1 或 R4 的网络管理员添加了一个新的联合体定义 x2，其中包括组织 R2 和 r3。这将用于为 x2 定义新的通道。

Notice that the network now has two consortia defined: X1 for organizations R1 and R2 and X2 for organizations R2 and R3. Consortium X2 has been introduced in order to be able to create a new channel for R2 and R3.

请注意，网络现在定义了两个联合体：x1 用于组织 r1 和 r2，x2 用于组织 r2 和 r3。为了能够为 R2 和 R3 创建新的通道，引入了联合体 X2。

A new channel can only be created by those organizations specifically identified in the network configuration policy, NC4, as having the appropriate rights to do so, i.e. R1 or R4. This is an example of a policy which separates organizations that can manage resources at the network level versus those who can manage resources at the channel level. Seeing these policies at work helps us understand why Hyperledger Fabric has a sophisticated tiered policy structure.

只有在网络配置策略 NC4 中明确标识的具有相应权限（即 R1 或 R4）的组织才能创建新的通道。这是一个策略示例，它将可以在网络级别管理资源的组织与可以在渠道级别管理资源的组织分开。在工作中看到这些策略有助于我们理解为什么 Hyperledger 结构具有复杂

的分层策略结构。

In practice, consortium definition X2 has been added to the network configuration NC4. We discuss the exact mechanics of this operation elsewhere in the documentation.

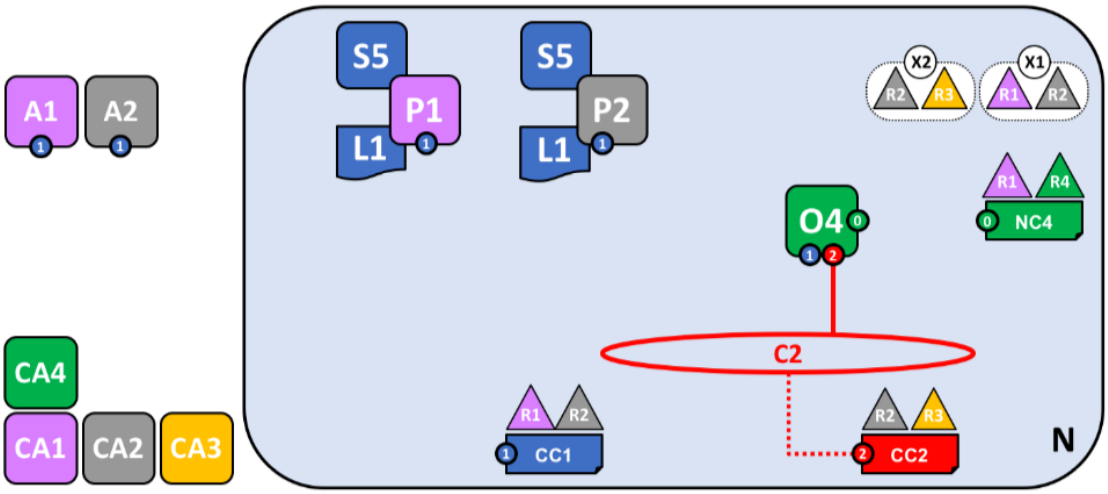
实际上，联合体定义 x2 已添加到网络配置 nc4 中。我们将在文档的其他地方讨论此操作的确切机制。

## 12、Adding a new channel

## 12、添加新通道

Let's now use this new consortium definition, X2, to create a new channel, C2. To help reinforce your understanding of the simpler channel notation, we've used both visual styles - channel C1 is represented with blue circular end points, whereas channel C2 is represented with red connecting lines:

现在让我们使用这个新的联合定义 x2 来创建一个新的通道 c2。为了加强您对简单通道符号的理解，我们使用了两种视觉样式——通道 C1 用蓝色圆形端点表示，而通道 C2 用红色连接线表示：




A new channel C2 has been created for R2 and R3 using consortium definition X2. The channel has a channel configuration CC2, completely separate to the network configuration NC4, and the channel configuration CC1. Channel C2 is managed by R2 and R3 who have equal rights over C2 as defined by a policy in CC2. R1 and R4 have no rights defined in CC2 whatsoever.

已使用联合体定义 x2 为 r2 和 r3 创建了新的信道 c2。通道具有通道配置 CC2，与网络配置 NC4 和通道配置 CC1 完全分离。渠道 C2 由 R2 和 R3 管理，他们对 C2 具有同等权利，如 CC2 中的政策所定义。R1 和 R4 没有 CC2 中定义的任何权利。

The channel C2 provides a private communications mechanism for the consortium X2. Again, notice how organizations united in a consortium are what form channels. The channel configuration CC2 now contains the policies that govern channel resources, assigning management rights to organizations R2 and R3 over channel C2. It is managed exclusively by R2 and R3; R1 and R4 have no power in channel C2. For example, channel configuration CC2 can subsequently be updated to add

organizations to support network growth, but this can only be done by R2 or R3.

信道 C2 为联合体 X2 提供了一个专用通信机制。同样，请注意组织如何联合在一个联合体中是什么形式的渠道。通道配置 cc2 现在包含管理通道资源的策略，通过通道 c2 向组织 r2 和 r3 分配管理权限。它完全由 R2 和 R3 管理；R1 和 R4 在信道 C2 中没有电源。例如，随后可以更新通道配置 cc2 以添加组织以支持网络增长，但这只能由 r2 或 r3 完成。

Note how the channel configurations CC1 and CC2 remain completely separate from each other, and completely separate from the network configuration, NC4. Again we' re seeing the de-centralized nature of a Hyperledger Fabric network; once channel C2 has been created, it is managed by organizations R2 and R3 independently to other network elements. Channel policies always remain separate from each other and can only be changed by the organizations authorized to do so in the channel.

注意信道配置 CC1 和 CC2 如何保持完全分离，并与网络配置 NC4 完全分离。我们再次看到了超级账本结构网络的非集中性；一旦创建了通道 C2，它就由组织 r2 和 r3 独立于其他网络元素进行管理。渠道政策始终保持相互独立，并且只能由授权在渠道中这样做的组织更改。

As the network and channels evolve, so will the network and channel configurations. There is a process by which this is accomplished in a controlled manner - involving configuration transactions which capture the change to these configurations. Every configuration change results in a new configuration block transaction being generated, and later in this topic, we' ll see how these blocks are validated and accepted to create updated network and channel configurations respectively.

随着网络和渠道的发展，网络和渠道的配置也将随之发展。这是一个以可控方式完成的过程，涉及捕获对这些配置的更改的配置事务。每次配置更改都会生成一个新的配置块事务，稍后在本主题中，我们将看到如何验证和接受这些块来分别创建更新的网络和通道配置。

**Network and channel configurations**

**网络和通道配置**

Throughout our sample network, we see the importance of network and channel configurations. These configurations are important because they encapsulate the policies agreed by the network members, which provide a shared reference for controlling access to network resources. Network and channel configurations also contain facts about the network and channel composition, such as the name of consortia and its organizations.

在我们的示例网络中，我们看到了网络和通道配置的重要性。这些配置很重要，因为它们封装了网络成员同意的策略，这些策略为控制对网络资源的访问提供了共享参考。网络和通道配置还包含有关网络和通道组成的事实，例如联合体及其组织的名称。

For example, when the network is first formed using the ordering service node O4, its behaviour is governed by the network configuration NC4. The initial configuration of NC4 only contains policies that permit organization R4 to manage network resources. NC4 is subsequently updated to also allow R1 to manage network resources. Once this change is made, any administrator from organization R1 or R4 that connects to O4 will have network management rights because that is what the policy in the network configuration NC4 permits. Internally, each node in

the ordering service records each channel in the network configuration, so that there is a record of each channel created, at the network level.

例如，当首次使用订购服务节点 O4 形成网络时，其行为由网络配置 NC4 控制。NC4 的初始配置仅包含允许组织 R4 管理网络资源的策略。NC4 随后更新，也允许 R1 管理网络资源。一旦进行了此更改，组织 R1 或 R4 中连接到 O4 的任何管理员都将拥有网络管理权限，因为这是网络配置 NC4 中的策略所允许的。在内部，订购服务中的每个节点记录网络配置中的每个通道，以便在网络级别上创建每个通道的记录。

It means that although ordering service node O4 is the actor that created consortia X1 and X2 and channels C1 and C2, the intelligence of the network is contained in the network configuration NC4 that O4 is obeying. As long as O4 behaves as a good actor, and correctly implements the policies defined in NC4 whenever it is dealing with network resources, our network will behave as all organizations have agreed. In many ways NC4 can be considered more important than O4 because, ultimately, it controls network access.

这意味着，尽管订购服务节点 O4 是创建联盟 x1 和 x2 以及通道 C1 和 C2 的参与者，但网络的智能包含在 O4 所遵循的网络配置 NC4 中。只要 O4 作为一个好的参与者，并且在处理网络资源时正确地执行 NC4 中定义的策略，我们的网络将按照所有组织都同意的方式运行。在许多方面，NC4 比 O4 更重要，因为它最终控制着网络访问。

The same principles apply for channel configurations with respect to peers. In our network, P1 and P2 are likewise good actors. When peer nodes P1 and P2 are interacting with client applications A1 or A2 they are each using the policies defined within channel configuration CC1 to control access to the channel C1 resources.

同样的原则也适用于相对于对等端的信道配置。在我们的网络中，p1 和 p2 同样是优秀的演员。当对等节点 p1 和 p2 与客户端应用程序 a1 或 a2 交互时，它们各自使用通道配置 cc1 中定义的策略来控制对通道 c1 资源的访问。

For example, if A1 wants to access the smart contract chaincode S5 on peer nodes P1 or P2, each peer node uses its copy of CC1 to determine the operations that A1 can perform. For example, A1 may be permitted to read or write data from the ledger L1 according to policies defined in CC1. We’ll see later the same pattern for actors in channel and its channel configuration CC2. Again, we can see that while the peers and applications are critical actors in the network, their behaviour in a channel is dictated more by the channel configuration policy than any other factor.

例如，如果 A1 想要访问对等节点 P1 或 P2 上的智能合约链代码 S5，则每个对等节点使用其 CC1 副本来确定 A1 可以执行的操作。例如，可以允许 A1 根据 CC1 中定义的策略从分类帐 L1 中读取或写入数据。稍后我们将看到通道中参与者的相同模式及其通道配置 cc2。同样，我们可以看到，虽然对等端和应用程序是网络中的关键参与者，但它们在通道中的行为更多地取决于通道配置策略，而不是任何其他因素。

Finally, it is helpful to understand how network and channel configurations are physically realized. We can see that network and channel configurations are logically singular – there is one for the network, and one for each channel. This is important; every component that accesses the network or the channel must have a shared understanding of the permissions granted to different organizations.

最后，它有助于理解网络和通道配置是如何在物理上实现的。我们可以看到网络和通道配置在逻辑上是单一的——网络有一个，每个通道有一个。这一点很重要；访问网络或通道的每个组件都必须共享对授予不同组织的权限的理解。

Even though there is logically a single configuration, it is actually replicated and kept consistent by every node that forms the network or channel. For example, in our network peer nodes P1 and P2 both have a copy of channel configuration CC1, and by the time the network is fully complete, peer nodes P2 and P3 will both have a copy of channel configuration CC2. Similarly ordering service node O4 has a copy of the network configuration, but in a multi-node configuration, every ordering service node will have its own copy of the network configuration.

即使在逻辑上只有一个配置，它实际上也是由构成网络或通道的每个节点复制并保持一致的。例如，在我们的网络中，对等节点 p1 和 p2 都有通道配置 cc1 的副本，当网络完全完成时，对等节点 p2 和 p3 都将有通道配置 cc2 的副本。同样，订购服务节点 O4 有一个网络配置的副本，但在多节点配置中，每个订购服务节点都有自己的网络配置副本。

Both network and channel configurations are kept consistent using the same blockchain technology that is used for user transactions - but for configuration transactions. To change a network or channel configuration, an administrator must submit a configuration transaction to change the network or channel configuration. It must be signed by the organizations identified in the appropriate policy as being responsible for configuration change. This policy is called the mod_policy and we'll discuss it later.

网络和渠道配置使用相同的区块链技术保持一致，用于用户交易，但用于配置交易。要更改网络或通道配置，管理员必须提交配置事务以更改网络或通道配置。必须由相应策略中确定负责配置更改的组织签署。这项政策被称为“国防部政策”，我们稍后再讨论。

Indeed, the ordering service nodes operate a mini-blockchain, connected via the system channel we mentioned earlier. Using the system channel ordering service nodes distribute network configuration transactions. These transactions are used to co-operatively maintain a consistent copy of the network configuration at each ordering service node. In a similar way, peer nodes in an application channel can distribute channel configuration transactions. Likewise, these transactions are used to maintain a consistent copy of the channel configuration at each peer node.

实际上，订购服务节点运行一个小型区块链，通过我们前面提到的系统通道连接。使用系统通道订购服务节点分发网络配置事务。这些事务用于在每个订购服务节点上合作维护网络配置的一致副本。以类似的方式，应用程序通道中的对等节点可以分发通道配置事务。同样，这些事务用于在每个对等节点上维护通道配置的一致副本。

This balance between objects that are logically singular, by being physically distributed is a common pattern in Hyperledger Fabric. Objects like network configurations, that are logically single, turn out to be physically replicated among a set of ordering services nodes for example. We also see it with channel configurations, ledgers, and to some extent smart contracts which are installed in multiple places but whose interfaces exist logically at the channel level. It's a pattern you see repeated time and again in Hyperledger Fabric, and

enables Hyperledger Fabric to be both de-centralized and yet manageable at the same time.

通过物理分布，逻辑上奇异的对象之间的这种平衡是 Hyperledger Fabric 中的一种常见模式。例如，逻辑上单一的网络配置之类的对象实际上是在一组订购服务节点之间进行物理复制的。我们还看到它与通道配置、分类账以及在某种程度上安装在多个位置但其接口在通道级别逻辑上存在的智能合约。这是一种在 Hyperledger Fabric 中反复出现的模式，它使 Hyperledger 结构既不集中，又可同时管理。

## 13、Adding another peer

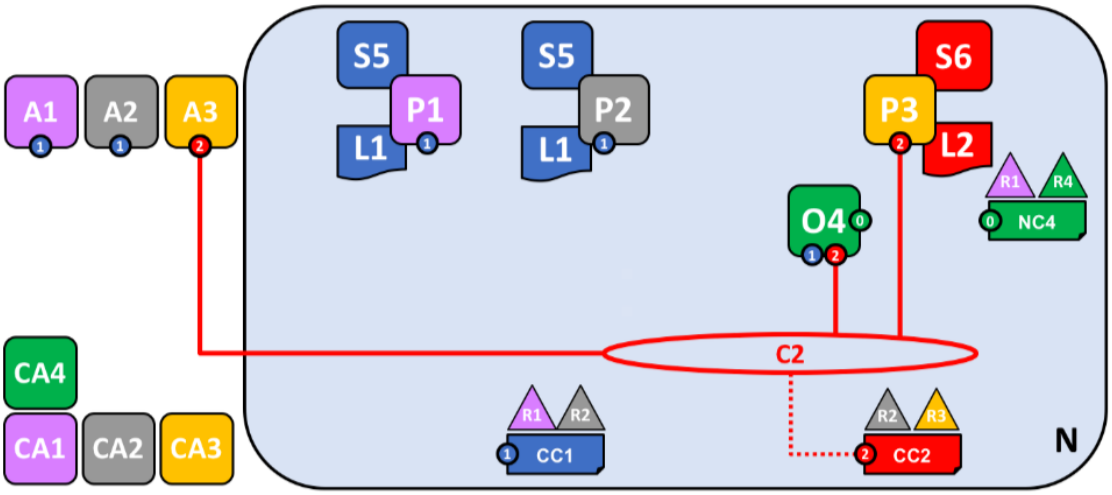
## 13、添加另一个节点

Now that organization R3 is able to fully participate in channel C2, let's add its infrastructure components to the channel. Rather than do this one component at a time, we're going to add a peer, its local copy of a ledger, a smart contract and a client application all at once!

既然组织 r3 能够完全参与渠道 c2，那么让我们将其基础设施组件添加到渠道中。与其一次只做一个组件，我们还不如一次添加一个对等点，一个分类账的本地副本，一个智能合约和一个客户端应用程序！

Let's see the network with organization R3's components added:

让我们看看添加了组织 r3 组件的网络：




The diagram shows the facts relating to channels C1 and C2 in the network N as follows: Client applications A1 and A2 can use channel C1 for communication with peers P1 and P2, and ordering service O4; client applications A3 can use channel C2 for communication with peer P3 and ordering service O4. Ordering service O4 can make use of the communication services of channels C1 and C2. Channel configuration CC1 applies to channel C1, CC2 applies to channel C2.

图中显示了与网络 n 中通道 c1 和 c2 相关的事实，如下所示：客户端应用程序 a1 和 a2 可以使用通道 c1 与对等端 p1 和 p2 通信，以及订购服务 o4；客户端应用程序 a3 可以使用通道 c2 与对等端 p3 和订购服务 o4 通信。订购服务 O4 可以利用信道 C1 和 C2 的通信服务。信道配置 CC1 适用于信道 C1，CC2 适用于信道 C2。

First of all, notice that because peer node P3 is connected to channel C2, it has a different ledger - L2 - to those peer nodes using channel C1. The ledger

L2 is effectively scoped to channel C2. The ledger L1 is completely separate; it is scoped to channel C1. This makes sense – the purpose of the channel C2 is to provide private communications between the members of the consortium X2, and the ledger L2 is the private store for their transactions.

首先，请注意，由于对等节点 p3 连接到通道 c2，因此使用通道 c1 的对等节点具有不同的分类账（12）。分类帐 L2 的有效范围为通道 C2。分类帐 11 是完全独立的；它的作用域是通道 c1。这是有意义的——渠道 C2 的目的是在联合体 X2 成员之间提供私人通信，而分类账 L2 是他们交易的私人存储。

In a similar way, the smart contract S6, installed on peer node P3, and instantiated on channel C2, is used to provide controlled access to ledger L2. Application A3 can now use channel C2 to invoke the services provided by smart contract S6 to generate transactions that can be accepted onto every copy of the ledger L2 in the network.

以类似的方式，安装在对等节点 P3 上并在通道 C2 上实例化的智能合约 S6 用于提供对分类账 L2 的受控访问。应用程序 a3 现在可以使用通道 c2 调用智能合约 s6 提供的服务，生成可以在网络中的每个分类账 12 副本上接受的交易。

At this point in time, we have a single network that has two completely separate channels defined within it. These channels provide independently managed facilities for organizations to transact with each other. Again, this is de-centralization at work; we have a balance between control and autonomy. This is achieved through policies which are applied to channels which are controlled by, and affect, different organizations.

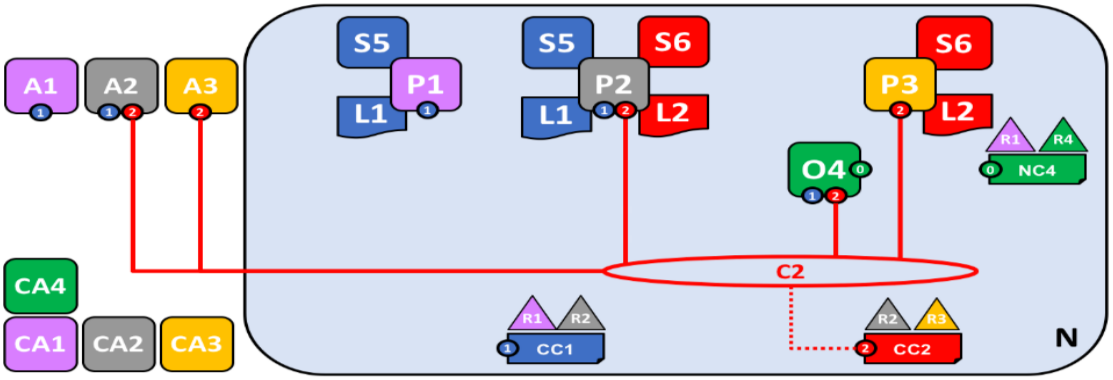
此时，我们有一个单独的网络，其中定义了两个完全独立的通道。这些渠道为组织之间的交易提供了独立管理的设施。同样，这是工作中的非集中化；我们在控制和自治之间保持平衡。这是通过适用于由不同组织控制和影响的渠道的政策实现的。

## 14、Joining a peer to multiple channels

## 14、一个节点加入多个通道

In this final stage of network development, let's return our focus to organization R2. We can exploit the fact that R2 is a member of both consortia X1 and X2 by joining it to multiple channels:

在网络开发的最后阶段，让我们将重点放在组织 R2 上。我们可以利用 r2 是联合体 x1 和 x2 成员的事实，将其加入多个渠道：



The diagram shows the facts relating to channels C1 and C2 in the network N as follows: Client applications A1 can use channel C1 for communication with peers P1 and P2, and ordering service O4; client application A2 can use channel C1 for communication with peers P1 and P2 and channel C2 for communication with peers P2 and P3 and ordering service O4; client application A3 can use channel C2 for communication with peer P3 and P2 and ordering service O4. Ordering service O4 can make use of the communication services of channels C1 and C2. Channel configuration CC1 applies to channel C1, CC2 applies to channel C2.

图中显示了与网络 n 中通道 c1 和 c2 相关的事实：客户端应用程序 a1 可以使用通道 c1 与对等端 p1 和 p2 通信，以及订购服务 o4；客户端应用程序 a2 可以使用通道 c1 与对等端 p1 和 p2 通信，通道 c2 与对等端 p2 和 p3 通信，以及订购服务。O4；客户机应用程序 A3 可以使用通道 C2 与对等机 P3 和 P2 以及订购服务 O4 进行通信。订购服务 O4 可以利用信道 C1 和 C2 的通信服务。信道配置 CC1 适用于信道 C1，CC2 适用于信道 C2。

We can see that R2 is a special organization in the network, because it is the only organization that is a member of two application channels! It is able to transact with organization R1 on channel C1, while at the same time it can also transact with organization R3 on a different channel, C2.

我们可以看到，R2 是网络中的一个特殊组织，因为它是两个应用渠道中唯一的一个成员组织！它能够在通道 C1 上与组织 R1 进行交易，同时也可以在不同通道 C2 上与组织 R3 进行交易。

Notice how peer node P2 has smart contract S5 installed for channel C1 and smart contract S6 installed for channel C2. Peer node P2 is a full member of both channels at the same time via different smart contracts for different ledgers.

注意对等节点 p2 如何为通道 c1 安装智能合约 s5，为通道 c2 安装智能合约 s6。对等节点 p2 通过不同分类账的不同智能合约同时成为两个渠道的完整成员。

This is a very powerful concept – channels provide both a mechanism for the separation of organizations, and a mechanism for collaboration between organizations. All the while, this infrastructure is provided by, and shared between, a set of independent organizations.

这是一个非常强大的概念——渠道既提供了组织分离的机制，也提供了组织间协作的机制。一直以来，这个基础设施都是由一组独立的组织提供的，并在它们之间共享。

It is also important to note that peer node P2's behaviour is controlled very differently depending upon the channel in which it is transacting. Specifically, the policies contained in channel configuration CC1 dictate the operations available to P2 when it is transacting in channel C1, whereas it is the policies in channel configuration CC2 that control P2's behaviour in channel C2.

还需要注意的是，对等节点 p2 的行为根据其处理的通道而受到非常不同的控制。具体来说，通道配置 CC1 中包含的策略规定了当 P2 在通道 C1 中进行事务处理时，它可以对 P2 执行的操作，而控制通道 C2 中 P2 行为的是通道配置 CC2 中的策略。

Again, this is desirable – R2 and R1 agreed the rules for channel C1, whereas R2 and R3 agreed the rules for channel C2. These rules were captured in the respective channel policies – they can and must be used by every component

in a channel to enforce correct behaviour, as agreed.

同样，这是可取的——R2 和 R1 同意通道 C1 的规则，而 R2 和 R3 同意通道 C2 的规则。这些规则包含在各自的渠道政策中——它们可以并且必须由渠道中的每个组件使用，以强制执行约定的正确行为。

Similarly, we can see that client application A2 is now able to transact on channels C1 and C2. And likewise, it too will be governed by the policies in the appropriate channel configurations. As an aside, note that client application A2 and peer node P2 are using a mixed visual vocabulary - both lines and connections. You can see that they are equivalent; they are visual synonyms.

同样，我们可以看到客户机应用程序 A2 现在能够在通道 C1 和 C2 上进行交易。同样，它也将由适当通道配置中的策略控制。另外，请注意，客户机应用程序 A2 和对等节点 P2 使用的是混合的可视词汇表——行和连接。您可以看到它们是等价的；它们是可视同义词。

**The ordering service**

**订购服务**

The observant reader may notice that the ordering service node appears to be a centralized component; it was used to create the network initially, and connects to every channel in the network. Even though we added R1 and R4 to the network configuration policy NC4 which controls the orderer, the node was running on R4's infrastructure. In a world of de-centralization, this looks wrong!

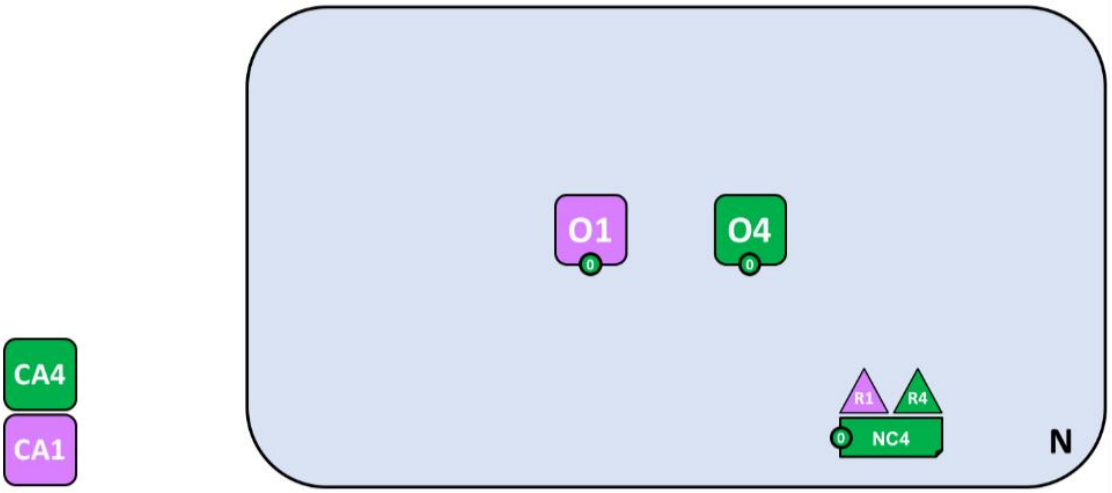
观察者可能会注意到订购服务节点似乎是一个集中的组件；它最初用于创建网络，并连接到网络中的每个通道。尽管我们在控制订购者的网络配置策略 NC4 中添加了 R1 和 R4，但是节点运行在 R4 的基础结构上。在一个非集中化的世界里，这看起来是错误的！

Don't worry! Our example network showed the simplest ordering service configuration to help you understand the idea of a network administration point. In fact, the ordering service can itself too be completely de-centralized! We mentioned earlier that an ordering service could be comprised of many individual nodes owned by different organizations, so let's see how that would be done in our sample network.

别担心！我们的示例网络显示了最简单的订购服务配置，以帮助您理解网络管理点的概念。实际上，订购服务本身也可以完全去中心化！我们前面提到订购服务可以由不同组织拥有的许多单个节点组成，所以让我们看看如何在示例网络中完成这一点。

Let's have a look at a more realistic ordering service node configuration:

让我们来看一个更现实的订购服务节点配置：




A multi-organization ordering service. The ordering service comprises ordering service nodes O1 and O4. O1 is provided by organization R1 and node O4 is provided by organization R4. The network configuration NC4 defines network resource permissions for actors from both organizations R1 and R4.

多组织订购服务。订购服务包括订购服务节点 O1 和 O4。o1 由组织 r1 提供，节点 o4 由组织 r4 提供。网络配置 NC4 为来自组织 R1 和 R4 的参与者定义网络资源权限。

We can see that this ordering service completely de-centralized - it runs in organization R1 and it runs in organization R4. The network configuration policy, NC4, permits R1 and R4 equal rights over network resources. Client applications and peer nodes from organizations R1 and R4 can manage network resources by connecting to either node O1 or node O4, because both nodes behave the same way, as defined by the policies in network configuration NC4. In practice, actors from a particular organization tend to use infrastructure provided by their home organization, but that's certainly not always the case.

我们可以看到这个订购服务完全不集中——它在组织 R1 中运行，在组织 R4 中运行。网络配置策略 NC4 允许 R1 和 R4 对网络资源享有平等的权利。来自组织 R1 和 R4 的客户端应用程序和对等节点可以通过连接到节点 O1 或节点 O4 来管理网络资源，因为这两个节点的行为与网络配置 NC4 中的策略定义的相同。在实践中，来自特定组织的参与者倾向于使用其所在组织提供的基础设施，但事实并非总是如此。

**De-centralized transaction distribution**

**非集中交易分配**

As well as being the management point for the network, the ordering service also provides another key facility - it is the distribution point for transactions. The ordering service is the component which gathers endorsed transactions from applications and orders them into transaction blocks, which are subsequently distributed to every peer node in the channel. At each of these committing peers, transactions are recorded, whether valid or invalid, and their local copy of the ledger updated appropriately.

除了作为网络的管理点，订购服务还提供了另一个关键设施——它是交易的分发点。订购服务是一个组件，它从应用程序中收集经过认可的事务，并将它们订购到事务块中，然后将事务块分发到通道中的每个对等节点。在这些提交对等方中的每一个，都会记录交易（无论是有效的还是无效的），并适当更新其分类账的本地副本。

Notice how the ordering service node O4 performs a very different role for the channel C1 than it does for the network N. When acting at the channel level, O4's role is to gather transactions and distribute blocks inside channel C1. It does this according to the policies defined in channel configuration CC1. In contrast, when acting at the network level, O4's role is to provide a management point for network resources according to the policies defined in network configuration NC4. Notice again how these roles are defined by different policies within the channel and network configurations respectively. This should reinforce to you the importance of declarative policy based configuration in Hyperledger Fabric. Policies both define, and are used to control, the agreed behaviours by each and every member of a consortium.

请注意，订购服务节点 O4 对通道 C1 的角色与对网络 N 的角色截然不同。当在通道级别执行操作时，O4 的角色是收集事务并在通道 C1 内分发块。它根据通道配置 CC1 中定义的策略执行此操作。相反，当在网络级别工作时，O4 的作用是根据网络配置 NC4 中定义的策略为网络资源提供管理点。再次注意这些角色是如何分别由通道和网络配置中的不同策略定义的。这应该加强在 Hyperledger 结构中基于声明性策略的配置的重要性。政策定义并用于控制联合体每个成员商定的行为。

We can see that the ordering service, like the other components in Hyperledger Fabric, is a fully de-centralized component. Whether acting as a network management point, or as a distributor of blocks in a channel, its nodes can be distributed as required throughout the multiple organizations in a network.

我们可以看到，订购服务与 Hyperledger 结构中的其他组件一样，是一个完全非集中的组件。无论是作为网络管理点，还是作为通道中块的分发者，其节点都可以根据需要分布在网络中的多个组织中。

**Changing policy**

**政策变化**

Throughout our exploration of the sample network, we've seen the importance of the policies to control the behaviour of the actors in the system. We've only discussed a few of the available policies, but there are many that can be declaratively defined to control every aspect of behaviour. These individual policies are discussed elsewhere in the documentation.

在我们对样本网络的探索中，我们已经看到了控制系统中参与者行为的政策的重要性。我们只讨论了一些可用的策略，但是有许多可以声明性地定义来控制行为的各个方面。这些个别政策将在文档的其他地方讨论。

Most importantly of all, Hyperledger Fabric provides a uniquely powerful policy that allows network and channel administrators to manage policy change itself! The underlying philosophy is that policy change is a constant, whether it occurs within or between organizations, or whether it is imposed by external regulators. For example, new organizations may join a channel, or existing organizations may have their permissions increased or decreased. Let's investigate a little more how change policy is implemented in Hyperledger Fabric.

最重要的是，Hyperledger Fabric 提供了一个独特的强大策略，允许网络和渠道管理员管理策略更改本身！基本的理念是，政策变化是一个常数，无论它发生在组织内部或组织之间，还是由外部监管机构强制实施。例如，新组织可以加入一个渠道，或者现有组织可以

增加或减少其权限。让我们进一步研究一下变更策略是如何在 Hyperledger Fabric 中实现的。

They key point of understanding is that policy change is managed by a policy within the policy itself. The modification policy, or mod_policy for short, is a first class policy within a network or channel configuration that manages change. Let's give two brief examples of how we've already used mod_policy to manage change in our network!

他们理解的关键点是，政策变更由政策本身内的政策管理。修改策略，简称 mod_策略，是管理更改的网络或通道配置中的一级策略。让我们举两个简单的例子来说明我们如何使用 mod_策略来管理我们网络中的变化！

The first example was when the network was initially set up. At this time, only organization R4 was allowed to manage the network. In practice, this was achieved by making R4 the only organization defined in the network configuration NC4 with permissions to network resources. Moreover, the mod_policy for NC4 only mentioned organization R4 - only R4 was allowed to change this configuration.

第一个例子是最初建立网络时。此时，仅允许组织 R4 管理网络。实际上，这是通过使 R4 成为网络配置 NC4 中定义的唯一具有网络资源权限的组织来实现的。此外，针对 NC4 的 mod_策略只提到了组织 r4——只允许 r4 更改此配置。

We then evolved the network N to also allow organization R1 to administer the network. R4 did this by adding R1 to the policies for channel creation and consortium creation. Because of this change, R1 was able to define the consortia X1 and X2, and create the channels C1 and C2. R1 had equal administrative rights over the channel and consortium policies in the network configuration.

然后，我们改进了网络 n，也允许组织 R1 管理网络。R4 通过在通道创建和联合体创建的策略中添加 R1 来实现这一点。由于这种变化，R1 能够定义联合体 x1 和 x2，并创建通道 c1 和 c2。R1 在网络配置中对通道和联合体策略拥有平等的管理权。

R4 however, could grant even more power over the network configuration to R1! R4 could add R1 to the mod_policy such that R1 would be able to manage change of the network policy too.

但是，R4 可以通过网络配置向 R1 授予更大的功率！R4 可以将 R1 添加到 mod_策略中，这样 R1 也可以管理网络策略的更改。

This second power is much more powerful than the first, because now R1 now has full control over the network configuration NC4! This means that R1 can, in principle remove R4's management rights from the network. In practice, R4 would configure the mod_policy such that R4 would need to also approve the change, or that all organizations in the mod_policy would have to approve the change. There's lots of flexibility to make the mod_policy as sophisticated as it needs to be to support whatever change process is required.

第二个功率比第一个功率大得多，因为现在 R1 可以完全控制网络配置 NC4！这意味着，原则上，R1 可以从网络中删除 R4 的管理权限。实际上，R4 将配置 mod_策略，以便 r4 也需要批准更改，或者 mod_策略中的所有组织都必须批准更改。有很大的灵活性，使国防部的政策像它需要的那样复杂，以支持任何需要的变革过程。

This is mod_policy at work - it has allowed the graceful evolution of a basic configuration into a sophisticated one. All the time this has occurred

with the agreement of all organization involved. The mod_policy behaves like every other policy inside a network or channel configuration; it defines a set of organizations that are allowed to change the mod_policy itself.

这是在工作中的 mod_策略——它允许基本配置优雅地演变为复杂配置。一直以来，所有相关组织都同意这一点。mod_策略的行为类似于网络或通道配置中的其他策略；它定义了一组允许更改 mod_策略本身的组织。

We’ ve only scratched the surface of the power of policies and mod_policy in particular in this subsection. It is discussed at much more length in the policy topic, but for now let’ s return to our finished network!

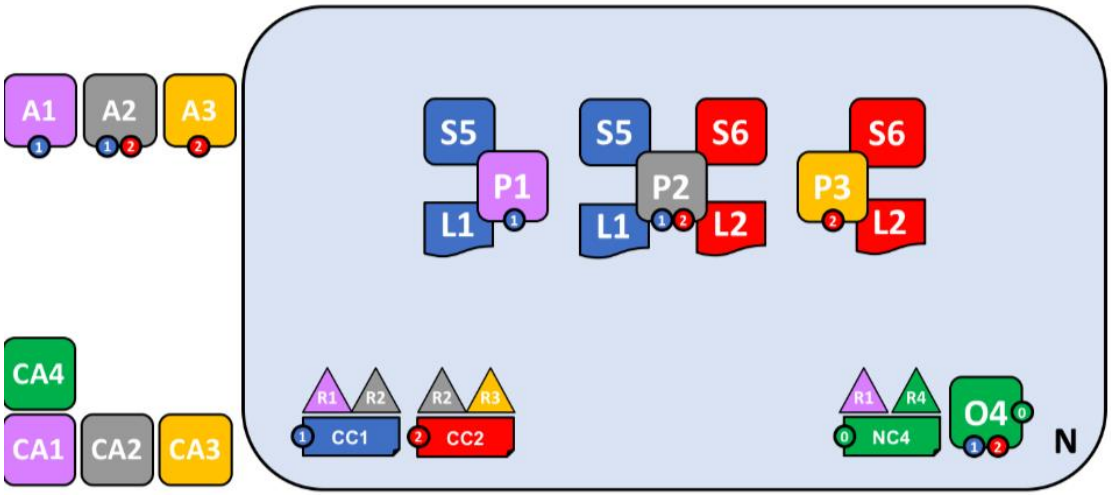
我们只触及了政策力量和国防政策的表面，特别是在本小节中。在政策主题中，我们将更详细地讨论这个问题，但现在让我们回到我们已经完成的网络！

## 15、Network fully formed

## 15、完全形成的网络

Let’ s recap what our network looks like using a consistent visual vocabulary. We’ ve re-organized it slightly using our more compact visual syntax, because it better accommodates larger topologies:

让我们用一致的视觉词汇回顾一下我们的网络是什么样子的。我们使用更紧凑的视觉语法对其进行了轻微的组织，因为它更好地适应更大的拓扑结构：




In this diagram we see that the Fabric blockchain network consists of two application channels and one ordering channel. The organizations R1 and R4 are responsible for the ordering channel, R1 and R2 are responsible for the blue application channel while R2 and R3 are responsible for the red application channel. Client applications A1 is an element of organization R1, and CA1 is its certificate authority. Note that peer P2 of organization R2 can use the communication facilities of the blue and the red application channel. Each application channel has its own channel configuration, in this case CC1 and CC2. The channel configuration of the system channel is part of the network configuration, NC4.

在这张图中，我们看到结构区块链网络由两个应用程序通道和一个订购通道组成。组织 R1 和 R4 负责订购渠道，R1 和 R2 负责蓝色应用渠道，R2 和 R3 负责红色应用渠道。客户机

应用程序 A1 是组织 R1 的一个元素，CA1 是其证书颁发机构。请注意，组织 r2 的对等机 p2 可以使用蓝色和红色应用程序通道的通信设施。每个应用程序通道都有自己的通道配置，在本例中是 CC1 和 CC2。系统通道的通道配置是网络配置 NC4 的一部分。

We’re at the end of our conceptual journey to build a sample Hyperledger Fabric blockchain network. We’ve created a four organization network with two channels and three peer nodes, with two smart contracts and an ordering service. It is supported by four certificate authorities. It provides ledger and smart contract services to three client applications, who can interact with it via the two channels. Take a moment to look through the details of the network in the diagram, and feel free to read back through the topic to reinforce your knowledge, or go to a more detailed topic.

我们的概念之旅即将结束，我们将构建一个样本的超级账本结构区块链网络。我们已经创建了一个具有两个通道和三个对等节点、两个智能合约和一个订购服务的四组织网络。它由四个证书颁发机构支持。它为三个客户应用程序提供分类帐和智能合约服务，客户应用程序可以通过两个渠道与之交互。花点时间浏览一下图中网络的细节，然后随意阅读主题以增强您的知识，或者转到更详细的主题。

**Summary of network components**

**网络组件摘要**

Here’s a quick summary of the network components we’ve discussed:

以下是我们讨论过的网络组件的快速摘要：

- Ledger. One per channel. Comprised of the Blockchain and the World state
- Smart contract (aka chaincode)
- Peer nodes
- Ordering service
- Channel
- Certificate Authority
- 分类帐。每个通道一个。由区块链和世界国家组成
- 智能合约（又名链码）
- 对等节点
- 订购服务
- 通道
- 证书颁发机构

## 16、Network summary

## 16、网络综述

In this topic, we’ve seen how different organizations share their infrastructure to provide an integrated Hyperledger Fabric blockchain network. We’ve seen how the collective infrastructure can be organized into channels that provide private communications mechanisms that are independently managed. We’ve seen how actors such as client applications, administrators, peers and

orderers are identified as being from different organizations by their use of certificates from their respective certificate authorities. And in turn, we've seen the importance of policy to define the agreed permissions that these organizational actors have over network and channel resources.

在本主题中,我们看到了不同的组织如何共享其基础设施以提供集成的超级账本结构区块链网络。我们已经看到了如何将集体基础设施组织成提供独立管理的私人通信机制的渠道。我们已经看到了如何通过使用来自各自证书颁发机构的证书来识别来自不同组织的参与者，如客户端应用程序、管理员、对等方和订购方。反过来，我们也看到了政策的重要性，它定义了这些组织参与者通过网络和渠道资源所拥有的商定权限。

# 五、Identity

# 五、身份

## 1、What is an Identity?

## 1、什么是身份

The different actors in a blockchain network include peers, orderers, client applications, administrators and more. Each of these actors — active elements inside or outside a network able to consume services — has a digital identity encapsulated in an X.509 digital certificate. These identities really matter because they determine the exact permissions over resources and access to information that actors have in a blockchain network.

区块链网络中的不同参与者包括对等方、订购方、客户机应用程序、管理员等等。这些参与者中的每一个——网络内或网络外能够使用服务的活动元素——都有一个封装在X.509数字证书中的数字标识。这些身份真的很重要，因为它们决定了参与者在区块链网络中对资源和信息访问的确切权限。

A digital identity furthermore has some additional attributes that Fabric uses to determine permissions, and it gives the union of an identity and the associated attributes a special name — principal. Principals are just like userIDs or groupIDs, but a little more flexible because they can include a wide range of properties of an actor's identity, such as the actor's organization, organizational unit, role or even the actor's specific identity. When we talk about principals, they are the properties which determine their permissions.

此外，数字标识还具有一些附加属性，Fabric使用这些属性来确定权限，它为标识和关联属性的联合提供了一个特殊的名称主体。主体与userid或groupid一样,但更灵活一些，因为它们可以包含参与者身份的广泛属性，例如参与者的组织、组织单位、角色，甚至参与者的特定身份。当我们谈论主体时，它们是决定其权限的属性。

For an identity to be verifiable, it must come from a trusted authority. A membership service provider (MSP) is how this is achieved in Fabric. More specifically, an MSP is a component that defines the rules that govern the valid identities for this organization. The default MSP implementation in Fabric uses X.509 certificates as identities, adopting a traditional Public Key Infrastructure (PKI) hierarchical model (more on PKI later).

要想验证身份，它必须来自可信的权威机构。会员服务提供商（MSP）就是在 Fabric 中实现这一点的方法。更具体地说，MSP 是一个组件，它定义了管理此组织的有效标识的规则。Fabric 中的默认 MSP 实现使用 X.509 证书作为标识，采用传统的公钥基础结构（PKI）层次模型（稍后将详细介绍 PKI）。

## 2、A Simple Scenario to Explain the Use of an Identity

## 2、一个简单的场景来解释身份的使用

Imagine that you visit a supermarket to buy some groceries. At the checkout you see a sign that says that only Visa, Mastercard and AMEX cards are accepted. If you try to pay with a different card — let's call it an "ImagineCard" — it doesn't matter whether the card is authentic and you have sufficient funds in your account. It will be not be accepted.

想象一下你去超市买东西。在结帐处，你看到一块牌子，上面写着只有维萨卡、万事达卡和美国运通卡可以接受。如果你尝试用另一张卡支付——我们称之为"想象卡"——这张卡是否真实，你的账户中是否有足够的资金并不重要。不接受。


Having a valid credit card is not enough — it must also be accepted by the store! PKIs and MSPs work together in the same way — a PKI provides a list of identities, and an MSP says which of these are members of a given organization that participates in the network.

拥有一张有效的信用卡是不够的-它也必须被商店接受！pkis 和 msp 以同样的方式工作——pki 提供了一个身份列表，msp 表示这些身份中的哪些人是参与网络的给定组织的成员。

PKI certificate authorities and MSPs provide a similar combination of functionalities. A PKI is like a card provider — it dispenses many different types of verifiable identities. An MSP, on the other hand, is like the list of card providers accepted by the store, determining which identities are the trusted members (actors) of the store payment network. MSPs turn verifiable identities into the members of a blockchain network.

PKI 证书颁发机构和 MSP 提供了类似的功能组合。一个 pki 就像一个卡提供商-它分配许多不同类型的可验证身份。另一方面，MSP 类似于商店接受的卡提供商列表，确定哪些身份是商店支付网络的受信任成员（参与者）。MSP 将可验证的身份转变为区块链网络的成员。

Let's drill into these concepts in a little more detail.

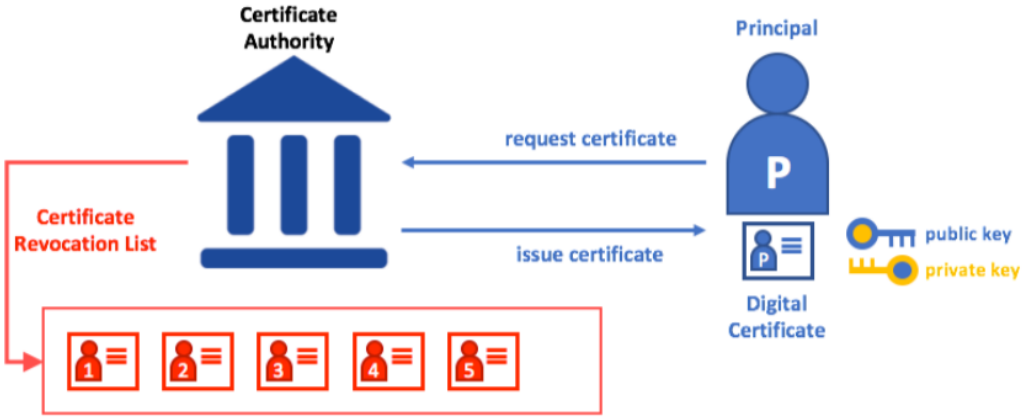
让我们更详细地了解一下这些概念。

## 3、What are PKIs?

## 3、什么是 PKIs

A public key infrastructure (PKI) is a collection of internet technologies that provides secure communications in a network. It's PKI that puts the S in HTTPS — and if you're reading this documentation on a web browser, you're probably using a PKI to make sure it comes from a verified source.

公钥基础设施（PKI）是提供网络中安全通信的 Internet 技术的集合。是 pki 将 s 放在了 https 中—如果您在 Web 浏览器上阅读此文档，那么您可能会使用 pki 来确保它来自经过验证的源。




The elements of Public Key Infrastructure (PKI). A PKI is comprised of Certificate Authorities who issue digital certificates to parties (e.g., users of a service, service provider), who then use them to authenticate themselves in the messages they exchange with their environment. A CA's Certificate Revocation List (CRL) constitutes a reference for the certificates that are no longer valid. Revocation of a certificate can happen for a number of reasons. For example, a certificate may be revoked because the cryptographic private material associated to the certificate has been exposed.

公钥基础设施（PKI）的元素。PKI 由向参与方（如服务用户、服务提供商）颁发数字证书的证书颁发机构组成，这些证书颁发机构随后使用它们在与环境交换的消息中进行身份验证。CA 的证书吊销列表（CRL）构成对不再有效的证书的引用。证书的吊销有多种原因。例如，证书可能会被吊销，因为与证书关联的加密私有材料已被公开。

Although a blockchain network is more than a communications network, it relies on the PKI standard to ensure secure communication between various network participants, and to ensure that messages posted on the blockchain are properly authenticated. It's therefore important to understand the basics of PKI and then why MSPs are so important.

虽然区块链网络不仅仅是一个通信网络，但它依赖于 PKI 标准来确保不同网络参与者之间的安全通信，并确保发布在区块链上的消息得到正确的认证。因此，了解 pki 的基础知识以及为什么 MSP 如此重要是很重要的。

There are four key elements to PKI:

PKI 有四个关键要素：

- Digital Certificates

- Public and Private Keys
- Certificate Authorities
- Certificate Revocation Lists
- 数字证书
- 公钥和私钥
- 证书颁发机构
- 证书吊销列表

Let’s quickly describe these PKI basics, and if you want to know more details, Wikipedia is a good place to start.

让我们快速描述一下这些 PKI 基础知识，如果您想了解更多细节，维基百科是一个很好的起点。

## 4、Digital Certificates

## 4、数字证书

A digital certificate is a document which holds a set of attributes relating to the holder of the certificate. The most common type of certificate is the one compliant with the X.509 standard, which allows the encoding of a party’s identifying details in its structure.

数字证书是一种文档，其中包含一组与证书持有人相关的属性。最常见的证书类型是符合 X.509 标准的证书，该标准允许对一方在其结构中的标识详细信息进行编码。

For example, Mary Morris in the Manufacturing Division of Mitchell Cars in Detroit, Michigan might have a digital certificate with a SUBJECT attribute of C=US, ST=Michigan, L=Detroit, O=Mitchell Cars, OU=Manufacturing, CN=Mary Morris /UID=123456. Mary’s certificate is similar to her government identity card — it provides information about Mary which she can use to prove key facts about her. There are many other attributes in an X.509 certificate, but let’s concentrate on just these for now.

例如，密歇根州底特律市 Mitchell Cars 制造部门的 Mary Morris 可能拥有主题属性为 c=us 、 st=Michigan 、 l=Detroit 、 o=Mitchell Cars 、 ou=Manufacturing 、 cn=Mary Morris/uid=123456 的数字证书。玛丽的证书和她的政府身份证很相似——它提供了玛丽的信息，她可以用这些信息来证明关于她的关键事实。在 X.509 证书中还有许多其他属性，但现在我们只关注这些属性。


A digital certificate describing a party called Mary Morris. Mary is the SUBJECT of the certificate, and the highlighted SUBJECT text shows key facts about Mary. The certificate also holds many more pieces of information, as you can see. Most importantly, Mary's public key is distributed within her certificate, whereas her private signing key is not. This signing key must be kept private.

描述一个叫玛丽·莫里斯的聚会的数字证书。玛丽是证书的主题，突出显示的主题文本显示了关于玛丽的关键事实。如您所见，证书还包含更多信息。最重要的是，玛丽的公钥分布在她的证书中，而她的私人签名密钥则不是。此签名密钥必须保密。

What is important is that all of Mary's attributes can be recorded using a mathematical technique called cryptography (literally, "secret writing") so that tampering will invalidate the certificate. Cryptography allows Mary to present her certificate to others to prove her identity so long as the other party trusts the certificate issuer, known as a Certificate Authority (CA). As long as the CA keeps certain cryptographic information securely (meaning, its own private signing key), anyone reading the certificate can be sure that the information about Mary has not been tampered with — it will always have those particular attributes for Mary Morris. Think of Mary's X.509 certificate as a digital identity card that is impossible to change.

重要的是，Mary 的所有属性都可以使用一种称为密码术（字面意思是"秘密书写"）的数学技术进行记录，这样篡改将使证书失效。只要另一方信任证书颁发者，即证书颁发机构（CA），密码学就允许玛丽向其他人出示证书以证明她的身份。只要 CA 安全地保存某些加密信息（也就是说，它自己的私人签名密钥），任何阅读证书的人都可以确保关于 Mary 的信息没有被篡改——它始终具有 Mary Morris 的那些特定属性。把玛丽的 X.509 证书想象成一张无法更改的数字身份证。

## 5、Authentication, Public keys, and Private Keys

## 5、身份验证、公钥和私钥

Authentication and message integrity are important concepts in secure communications. Authentication requires that parties who exchange messages are

assured of the identity that created a specific message. For a message to have “integrity” means that cannot have been modified during its transmission. For example, you might want to be sure you’re communicating with the real Mary Morris rather than an impersonator. Or if Mary has sent you a message, you might want to be sure that it hasn’t been tampered with by anyone else during transmission.

身份验证和消息完整性是安全通信中的重要概念。身份验证要求交换消息的各方确保创建特定消息的身份。对于具有“完整性”的消息，意味着在其传输过程中无法修改。例如，你可能想确定你在和真正的玛丽·莫里斯交流，而不是一个冒充者。或者，如果玛丽给你发了一条信息，你可能想确保它在传输过程中没有被其他人篡改。

Traditional authentication mechanisms rely on digital signatures that, as the name suggests, allow a party to digitally sign its messages. Digital signatures also provide guarantees on the integrity of the signed message.

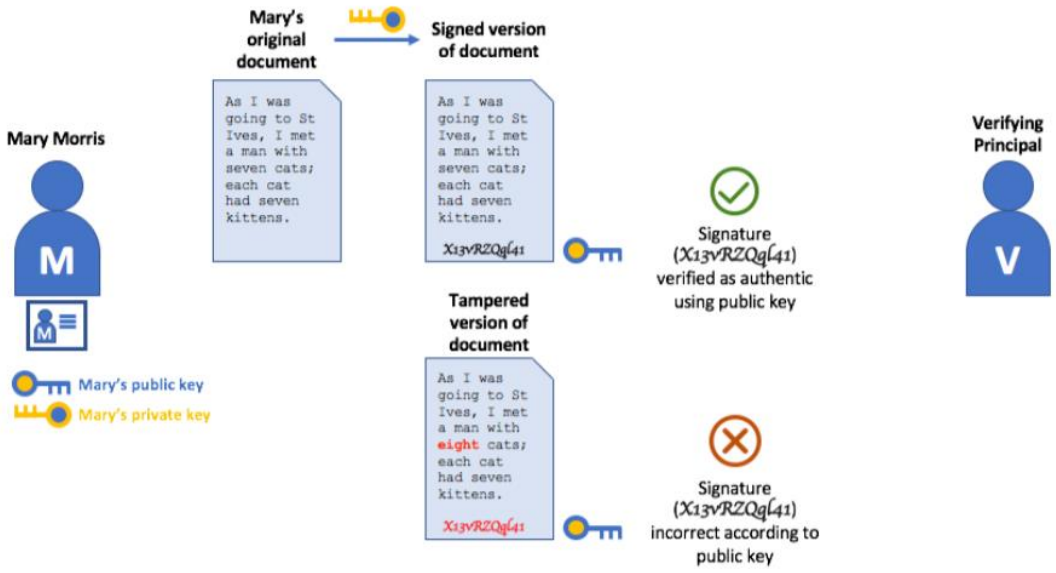
传统的认证机制依赖于数字签名，顾名思义，数字签名允许一方对其消息进行数字签名。数字签名还可以保证签名消息的完整性。

Technically speaking, digital signature mechanisms require each party to hold two cryptographically connected keys: a public key that is made widely available and acts as authentication anchor, and a private key that is used to produce digital signatures on messages. Recipients of digitally signed messages can verify the origin and integrity of a received message by checking that the attached signature is valid under the public key of the expected sender.

从技术上讲，数字签名机制要求各方持有两个密码学上连接的密钥：一个广泛可用并充当身份验证锚的公钥，以及一个用于在消息上生成数字签名的私钥。数字签名邮件的收件人可以通过检查所附签名在预期发件人的公钥下是否有效来验证接收邮件的来源和完整性。

The unique relationship between a private key and the respective public key is the cryptographic magic that makes secure communications possible. The unique mathematical relationship between the keys is such that the private key can be used to produce a signature on a message that only the corresponding public key can match, and only on the same message.

私钥和各自的公钥之间的唯一关系是使安全通信成为可能的加密魔法。密钥之间的唯一数学关系是，可以使用私钥在只有相应公钥可以匹配的消息上生成签名，并且只能在同一消息上生成签名。




In the example above, Mary uses her private key to sign the message. The signature can be verified by anyone who sees the signed message using her public key.

在上面的示例中，Mary 使用她的私钥来签署消息。任何人都可以使用她的公钥来验证签名消息。

## 6、Certificate Authorities

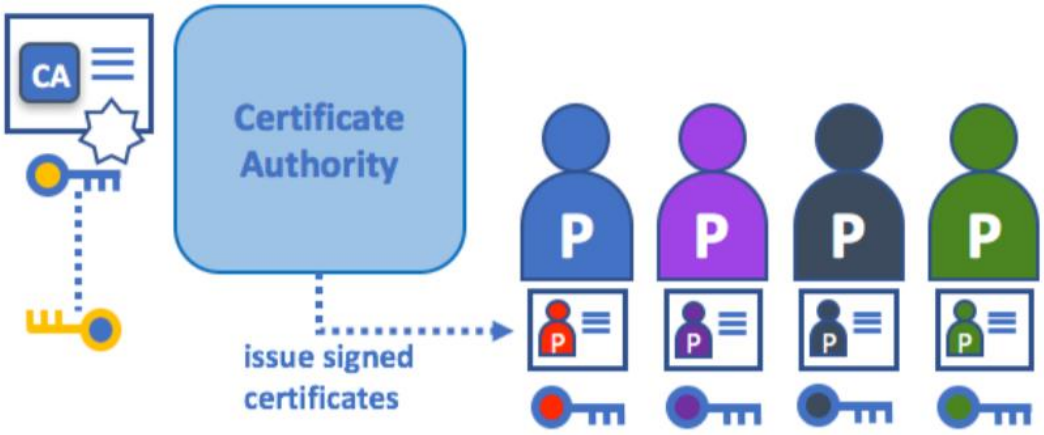
## 6、证书颁发机构

As you've seen, an actor or a node is able to participate in the blockchain network, via the means of a digital identity issued for it by an authority trusted by the system. In the most common case, digital identities (or simply identities) have the form of cryptographically validated digital certificates that comply with X.509 standard and are issued by a Certificate Authority (CA).

如您所见，参与者或节点能够通过系统信任的权威机构为其颁发的数字标识参与区块链网络。在最常见的情况下，数字标识（或简单的标识）具有符合 X.509 标准并由证书颁发机构（CA）颁发的加密验证数字证书的形式。

CAs are a common part of internet security protocols, and you've probably heard of some of the more popular ones: Symantec (originally Verisign), GeoTrust, DigiCert, GoDaddy, and Comodo, among others.

CA 是 Internet 安全协议的常见组成部分，您可能听说过一些比较流行的协议：Symantec（最初是 Verisign）、GeoTrust、Digicert、Godaddy 和 Comodo 等。




A Certificate Authority dispenses certificates to different actors. These certificates are digitally signed by the CA and bind together the actor's public key (and optionally with a comprehensive list of properties). As a result, if one trusts the CA (and knows its public key), it can trust that the specific actor is bound to the public key included in the certificate, and owns the included attributes, by validating the CA's signature on the actor's certificate.

证书颁发机构将证书分发给不同的参与者。这些证书由 CA 数字签名，并使用参与者的公钥（也可以选择使用完整的属性列表）将参与者绑定在一起。因此，如果信任 CA（并且知道其公钥），则可以通过验证 CA 在参与者证书上的签名来信任特定参与者绑定到证书中包含的公钥，并拥有包含的属性。

Certificates can be widely disseminated, as they do not include either the actors' nor the CA's private keys. As such they can be used as anchor of trusts for authenticating messages coming from different actors.

证书可以广泛传播，因为它们既不包括参与者的私钥，也不包括 CA 的私钥。因此，它们可以用作信任的锚，用于验证来自不同参与者的消息。

CAs also have a certificate, which they make widely available. This allows the consumers of identities issued by a given CA to verify them by checking that the certificate could only have been generated by the holder of the corresponding private key (the CA).

中科院也有一个证书，他们提供了广泛的使用。这允许给定 CA 颁发的身份的使用者通过检查证书是否只能由相应私钥（CA）的持有人生成来验证它们。

In a blockchain setting, every actor who wishes to interact with the network needs an identity. In this setting, you might say that one or more CAs can be used to define the members of an organization's from a digital perspective. It's the CA that provides the basis for an organization's actors to have a verifiable digital identity.

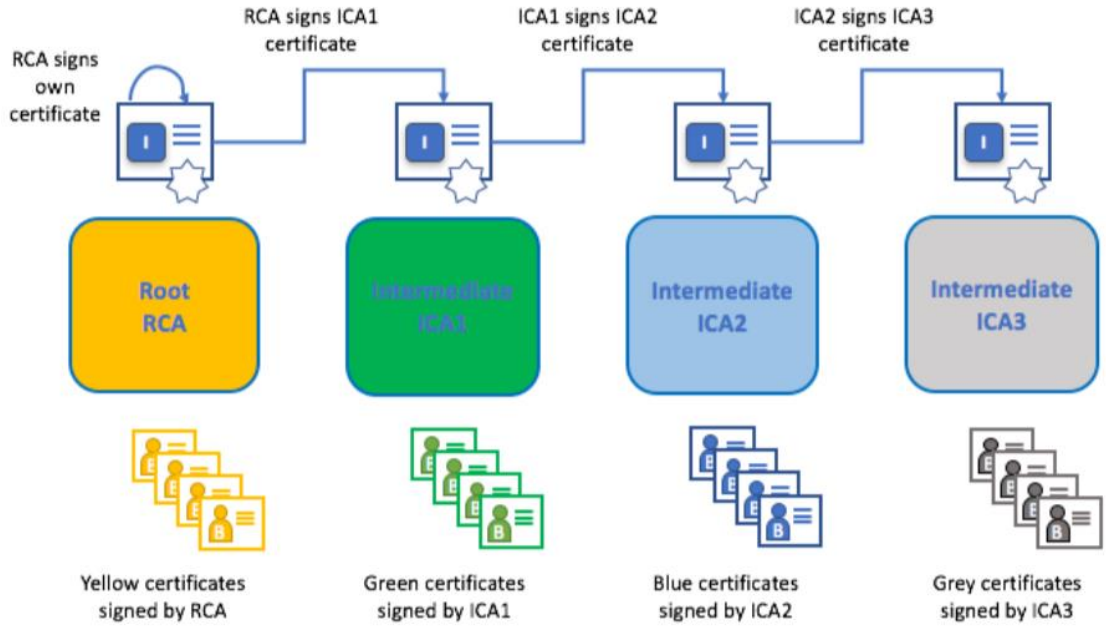
在区块链设置中，每个希望与网络交互的参与者都需要一个身份。在此设置中，您可能会说，可以使用一个或多个 CA 从数字角度定义组织的成员。正是 CA 为组织的参与者提供了一个可验证的数字身份的基础。

**Root CAs, Intermediate CAs and Chains of Trust**

**根 CA、中间 CA 和信任链**

CAs come in two flavors: Root CAs and Intermediate CAs. Because Root CAs (Symantec, Geotrust, etc) have to securely distribute hundreds of millions of certificates to internet users, it makes sense to spread this process out across what are called Intermediate CAs. These Intermediate CAs have their certificates issued by the root CA or another intermediate authority, allowing the establishment of a "chain of trust" for any certificate that is issued by any CA in the chain. This ability to track back to the Root CA not only allows the function of CAs to scale while still providing security — allowing organizations that consume certificates to use Intermediate CAs with confidence — it limits the exposure of the Root CA, which, if compromised, would endanger the entire chain of trust. If an Intermediate CA is compromised, on the other hand, there will be a much smaller exposure.

CAS 有两种形式：根 CA 和中间 CA。因为根 CA（Symantec、GeoTrust 等）必须安全地向 Internet 用户分发数亿个证书，所以有必要将此过程传播到称为中间 CA 的地方。这些中间 CA 具有由根 CA 或其他中间颁发机构颁发的证书，允许为链中任何 CA 颁发的任何证书建立"信任链"。这种追溯到根 CA 的能力不仅允许 CA 的功能在提供安全性的同时进行扩展——允许使用证书的组织自信地使用中间 CA——它限制了根 CA 的暴露，如果泄露，将危及整个信任链。另一方面，如果一个中间的钙被破坏，那么暴露量会小得多。




A chain of trust is established between a Root CA and a set of Intermediate CAs as long as the issuing CA for the certificate of each of these Intermediate CAs is either the Root CA itself or has a chain of trust to the Root CA.

在根 CA 和一组中间 CA 之间建立信任链，只要这些中间 CA 的证书的颁发 CA 是根 CA 本身或具有到根 CA 的信任链。

Intermediate CAs provide a huge amount of flexibility when it comes to the issuance of certificates across multiple organizations, and that's very helpful in a permissioned blockchain system (like Fabric). For example, you'll see that different organizations may use different Root CAs, or the same Root CA with

different Intermediate CAs — it really does depend on the needs of the network.

当涉及跨多个组织颁发证书时，中间 CA 提供了巨大的灵活性，这对于许可的区块链系统（如 Fabric）非常有帮助。例如，您将看到不同的组织可能使用不同的根 CA，或者使用不同中间 CA 的相同根 CA—这确实取决于网络的需要。

Fabric CA

Fabric CA

It’s because CAs are so important that Fabric provides a built-in CA component to allow you to create CAs in the blockchain networks you form. This component — known as Fabric CA is a private root CA provider capable of managing digital identities of Fabric participants that have the form of X.509 certificates. Because Fabric CA is a custom CA targeting the Root CA needs of Fabric, it is inherently not capable of providing SSL certificates for general/automatic use in browsers. However, because some CA must be used to manage identity (even in a test environment), Fabric CA can be used to provide and manage certificates. It is also possible — and fully appropriate — to use a public/commerical root or intermediate CA to provide identification.

这是因为 CA 非常重要，Fabric 提供了一个内置的 CA 组件，允许您在所形成的区块链网络中创建 CA。这个组件（称为 Fabric CA）是一个私有的根 CA 提供者，能够管理具有 X.509 证书形式的 Fabric 参与者的数字身份。因为 Fabric CA 是一个针对 Fabric 根 CA 需求的自定义 CA，所以它本质上不能为浏览器中的常规/自动使用提供 SSL 证书。但是，由于必须使用某些 CA 来管理身份（即使在测试环境中），因此可以使用结构 CA 来提供和管理证书。使用公共/商业根目录或中间 CA 来提供标识也是完全合适的。

If you’re interested, you can read a lot more about Fabric CA in the CA documentation section.

如果您感兴趣，可以在 CA 文档部分了解更多关于 Fabric CA 的信息。

## 7、Certificate Revocation Lists

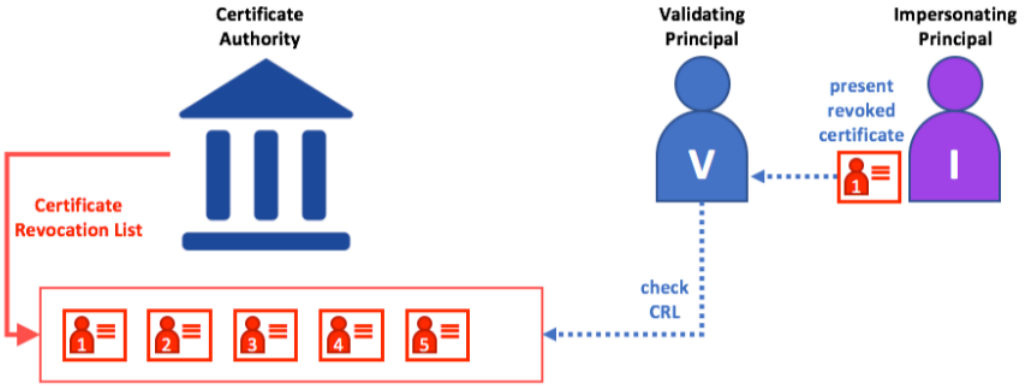
## 7、证书吊销列表

A Certificate Revocation List (CRL) is easy to understand — it’s just a list of references to certificates that a CA knows to be revoked for one reason or another. If you recall the store scenario, a CRL would be like a list of stolen credit cards.

证书撤销列表（crl）很容易理解——它只是一个 CA 知道由于某种原因被撤销的证书的引用列表。如果你回忆起商店场景，一个 CRL 就像一张被盗信用卡的列表。

When a third party wants to verify another party’s identity, it first checks the issuing CA’s CRL to make sure that the certificate has not been revoked. A verifier doesn’t have to check the CRL, but if they don’t they run the risk of accepting a compromised identity.

当第三方想要验证另一方的身份时，它首先检查颁发 CA 的 CRL，以确保证书没有被吊销。验证器不必检查 CRL，但是如果不检查，他们就有接受泄露身份的风险。




Using a CRL to check that a certificate is still valid. If an impersonator tries to pass a compromised digital certificate to a validating party, it can be first checked against the issuing CA's CRL to make sure it's not listed as no longer valid.

使用 CRL 检查证书是否仍然有效。如果模拟者试图将已损坏的数字证书传递给验证方，则可以首先根据颁发 CA 的 CRL 对其进行检查，以确保它不再被列为有效证书。

Note that a certificate being revoked is very different from a certificate expiring. Revoked certificates have not expired — they are, by every other measure, a fully valid certificate. For more in-depth information about CRLs, click here.

请注意，被吊销的证书与过期的证书非常不同。被吊销的证书尚未过期-根据其他衡量标准，它们是完全有效的证书。有关 CRL 的详细信息，请单击此处。

Now that you've seen how a PKI can provide verifiable identities through a chain of trust, the next step is to see how these identities can be used to represent the trusted members of a blockchain network. That's where a Membership Service Provider (MSP) comes into play — it identifies the parties who are the members of a given organization in the blockchain network.

既然您已经了解了 PKI 如何通过信任链提供可验证的身份，那么下一步就是了解如何使用这些身份来表示区块链网络的受信任成员。这就是会员服务提供商（MSP）发挥作用的地方——它识别出区块链网络中特定组织的成员。

To learn more about membership, check out the conceptual documentation on MSPs.

要了解更多关于成员身份的信息，请查看 MSP 的概念文档。

# 六、Membership

# 六、会员

If you've read through the documentation on identity you've seen how a PKI can provide verifiable identities through a chain of trust. Now let's see how these identities can be used to represent the trusted members of a blockchain network.

如果您已经阅读了有关身份的文档，那么您已经看到了 pki 如何通过信任链提供可验证的身份。现在让我们看看如何使用这些身份来表示区块链网络的可信成员。

This is where a Membership Service Provider (MSP) comes into play — it identifies which Root CAs and Intermediate CAs are trusted to define the members of a trust domain, e.g., an organization, either by listing the identities of their members, or by identifying which CAs are authorized to issue valid identities for their members, or — as will usually be the case — through a combination of both.

这就是会员服务提供商（MSP）发挥作用的地方——它通过列出成员的身份，或者通过确定授权哪个 CA 为成员颁发有效身份，或者根据需要，确定哪些根 CA 和中间 CA 受信任以定义信任域的成员，例如组织。通常是这样的-通过两者的结合。

The power of an MSP goes beyond simply listing who is a network participant or member of a channel. An MSP can identify specific roles an actor might play either within the scope of the organization the MSP represents (e.g., admins, or as members of a sub-organization group), and sets the basis for defining access privileges in the context of a network and channel (e.g., channel admins, readers, writers).

MSP 的功能不仅仅是列出谁是网络参与者或渠道成员。MSP 可以识别参与者在 MSP 所代表的组织范围内（例如，管理员或作为子组织组的成员）可能扮演的特定角色，并设置在网络和通道上下文中定义访问权限的基础（例如，通道管理员、读卡器、编写器）。

The configuration of an MSP is advertised to all the channels where members of the corresponding organization participate (in the form of a channel MSP). In addition to the channel MSP, peers, orderers, and clients also maintain a local MSP to authenticate member messages outside the context of a channel and to define the permissions over a particular component (who has the ability to install chaincode on a peer, for example).

MSP 的配置被公布到相应组织成员参与的所有通道（以通道 MSP 的形式）。除了通道 MSP 之外，对等方、订购方和客户机还维护本地 MSP，以在通道上下文之外对成员消息进行身份验证，并定义对特定组件（例如，具有在对等方上安装链码的能力）的权限。

In addition, an MSP can allow for the identification of a list of identities that have been revoked — as discussed in the Identity documentation — but we will talk about how that process also extends to an MSP.

此外，MSP 可以允许识别已撤销的标识列表（如标识文档中所述），但我们将讨论该过程如何扩展到 MSP。

We’ll talk more about local and channel MSPs in a moment. For now let’s see what MSPs do in general.

稍后我们将进一步讨论本地和频道 MSP。现在，让我们看看 MSP 通常做什么。

## 1、Mapping MSPs to Organizations

## 1、将 MSP 映射到组织

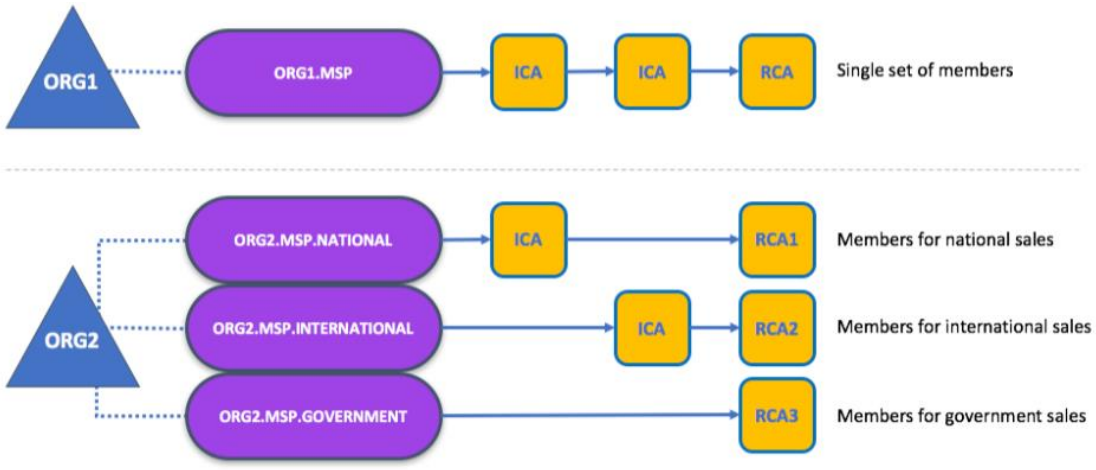
An organization is a managed group of members. This can be something as big as a multinational corporation or a small as a flower shop. What’s most important about organizations (or orgs) is that they manage their members under a single MSP. Note that this is different from the organization concept

defined in an X.509 certificate, which we'll talk about later.

组织是受管理的成员组。这可以是像跨国公司那样大的东西，也可以是像花店那样小的东西。组织（或组织）最重要的是，他们在一个 MSP 下管理自己的成员。请注意，这与 X.509 证书中定义的组织概念不同，稍后我们将讨论该证书。

The exclusive relationship between an organization and its MSP makes it sensible to name the MSP after the organization, a convention you'll find adopted in most policy configurations. For example, organization ORG1 would likely have an MSP called something like ORG1-MSP. In some cases an organization may require multiple membership groups — for example, where channels are used to perform very different business functions between organizations. In these cases it makes sense to have multiple MSPs and name them accordingly, e.g., ORG2-MSP-NATIONAL and ORG2-MSP-GOVERNMENT, reflecting the different membership roots of trust within ORG2 in the NATIONAL sales channel compared to the GOVERNMENT regulatory channel.

组织与其 MSP 之间的排他关系使得将 MSP 命名为组织之后更为明智，这是在大多数策略配置中都会采用的一种约定。例如，组织 org1 可能有一个名为 org1-msp 的 MSP。在某些情况下，一个组织可能需要多个成员组，例如，使用渠道在组织之间执行非常不同的业务功能。在这些情况下，有多个 MSP 并相应地命名是有意义的，例如，ORG2-MSP-National 和 ORG2-MSP-Government，这反映了与政府监管渠道相比，ORG2 在国家销售渠道中的不同成员信任根源。




Two different MSP configurations for an organization. The first configuration shows the typical relationship between an MSP and an organization — a single MSP defines the list of members of an organization. In the second configuration, different MSPs are used to represent different organizational groups with national, international, and governmental affiliation.

一个组织有两种不同的 MSP 配置。第一个配置显示了 MSP 和组织之间的典型关系——单个 MSP 定义了组织成员的列表。在第二种配置中，不同的 MSP 用于表示具有国家、国际和政府隶属关系的不同组织组。

**Organizational Units and MSPs**

**组织单位和 MSP**

An organization is often divided up into multiple organizational units

(OUs), each of which has a certain set of responsibilities. For example, the ORG1 organization might have both ORG1-MANUFACTURING and ORG1-DISTRIBUTION OUs to reflect these separate lines of business. When a CA issues X.509 certificates, the OU field in the certificate specifies the line of business to which the identity belongs.

一个组织通常被划分为多个组织单元（OU），每个组织单元都有一组特定的职责。例如，org1 组织可能同时拥有 org1-manufacturing 和 org1-distribution 组织单位，以反映这些单独的业务线。当 CA 颁发 X.509 证书时，证书中的 ou 字段指定标识所属的业务线。

We’ll see later how OUs can be helpful to control the parts of an organization who are considered to be the members of a blockchain network. For example, only identities from the ORG1-MANUFACTURING OU might be able to access a channel, whereas ORG1-DISTRIBUTION cannot.

稍后我们将看到，OU 如何有助于控制被认为是区块链网络成员的组织部分。例如，只有来自 ORG1-Manufacturing OU 的标识才能访问通道，而 ORG1-Distribution 则不能。

Finally, though this is a slight misuse of OUs, they can sometimes be used by different organizations in a consortium to distinguish each other. In such cases, the different organizations use the same Root CAs and Intermediate CAs for their chain of trust, but assign the OU field to identify members of each organization. We’ll also see how to configure MSPs to achieve this later.

最后，虽然这是对 OU 的轻微滥用，但有时不同的组织可以在一个联合体中使用它们来区分彼此。在这种情况下，不同的组织使用相同的根 CA 和中间 CA 作为其信任链，但分配 ou 字段来标识每个组织的成员。稍后我们还会看到如何配置 MSP 来实现这一点。

## 2、Local and Channel MSPs

## 2、本地和信道 MSP

MSPs appear in two places in a blockchain network: channel configuration (channel MSPs), and locally on an actor’s premise (local MSP). Local MSPs are defined for clients (users) and for nodes (peers and orderers). Node local MSPs define the permissions for that node (who the peer admins are, for example). The local MSPs of the users allow the user side to authenticate itself in its transactions as a member of a channel (e.g. in chaincode transactions), or as the owner of a specific role into the system (an org admin,for example, in configuration transactions).

MSP 出现在区块链网络中的两个位置：通道配置（通道 MSP）和本地参与者前提（本地 MSP）。本地 MSP 是为客户机（用户）和节点（对等机和订单机）定义的。节点本地 MSP 定义该节点的权限（例如，对等管理员是谁）。用户的本地 MSP 允许用户端在其事务中作为通道成员（例如，在链码事务中）或作为系统中特定角色的所有者（例如，在配置事务中）进行身份验证。

Every node and user must have a local MSP defined, as it defines who has administrative or participatory rights at that level (peer admins will not

necessarily be channel admins, and vice versa).

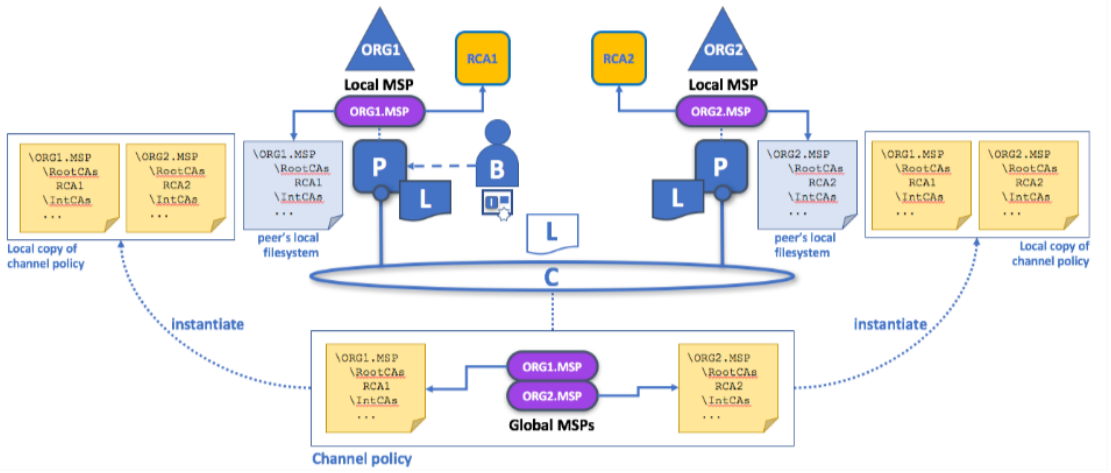
每个节点和用户都必须定义一个本地 MSP，因为它定义了谁在该级别上具有管理或参与权限（对等管理员不一定是通道管理员，反之亦然）。

In contrast, channel MSPs define administrative and participatory rights at the channel level. Every organization participating in a channel must have an MSP defined for it. Peers and orderers on a channel will all share the same view of channel MSPs, and will therefore be able to correctly authenticate the channel participants. This means that if an organization wishes to join the channel, an MSP incorporating the chain of trust for the organization's members would need to be included in the channel configuration. Otherwise transactions originating from this organization's identities will be rejected.

相反，渠道 MSP 在渠道级别定义了管理权和参与权。每个参与渠道的组织都必须为其定义 MSP。通道上的对等方和订购方都将共享通道 MSP 的相同视图，因此能够正确地验证通道参与者。这意味着，如果一个组织希望加入通道，那么需要在通道配置中包含一个包含该组织成员信任链的 MSP。否则，来自此组织标识的事务将被拒绝。

The key difference here between local and channel MSPs is not how they function — both turn identities into roles — but their scope.

本地和通道 MSP 之间的关键区别不在于它们的工作方式——两者都将身份转换为角色——而是它们的作用域。




Local and channel MSPs. The trust domain (e.g., the organization) of each peer is defined by the peer's local MSP, e.g., ORG1 or ORG2. Representation of an organization on a channel is achieved by adding the organization's MSP to the channel configuration. For example, the channel of this figure is managed by both ORG1 and ORG2. Similar principles apply for the network, orderers, and users, but these are not shown here for simplicity.

本地和信道 MSP。每个对等机的信任域（例如，组织）由对等机的本地 MSP（例如，ORG1 或 ORG2）定义。通过将组织的 MSP 添加到通道配置中，可以实现组织在通道上的表示。例如，这个图的通道由 org1 和 org2 管理。类似的原则也适用于网络、订购者和用户，但是为了简单起见，这里不显示这些原则。

You may find it helpful to see how local and channel MSPs are used by seeing what happens when a blockchain administrator installs and instantiates

a smart contract, as shown in the diagram above.

您可能会发现，通过查看区块链管理员安装和实例化智能合约时会发生什么情况，可以了解如何使用本地和渠道 MSP，如上图所示。

An administrator B connects to the peer with an identity issued by RCA1 and stored in their local MSP. When B tries to install a smart contract on the peer, the peer checks its local MSP, ORG1-MSP, to verify that the identity of B is indeed a member of ORG1. A successful verification will allow the install command to complete successfully. Subsequently, B wishes to instantiate the smart contract on the channel. Because this is a channel operation, all organizations on the channel must agree to it. Therefore, the peer must check the MSPs of the channel before it can successfully commit this command. (Other things must happen too, but concentrate on the above for now.)

管理员 B 使用 RCA1 颁发的标识连接到对等机，并存储在其本地 MSP 中。当 B 尝试在对等端上安装智能合约时，对等端会检查其本地 MSP org1-msp，以验证 B 的标识是否确实是 org1 的成员。成功的验证将允许安装命令成功完成。随后，B 希望在通道上实例化智能合约。因为这是一个渠道操作，所以渠道上的所有组织都必须同意它。因此，在成功提交此命令之前，对等端必须检查通道的 MSP。（其他事情也必须发生，但现在要专注于以上内容。）

Local MSPs are only defined on the file system of the node or user to which they apply. Therefore, physically and logically there is only one local MSP per node or user. However, as channel MSPs are available to all nodes in the channel, they are logically defined once in the channel configuration. However, a channel MSP is also instantiated on the file system of every node in the channel and kept synchronized via consensus. So while there is a copy of each channel MSP on the local file system of every node, logically a channel MSP resides on and is maintained by the channel or the network.

本地 MSP 仅在其应用的节点或用户的文件系统上定义。因此，在物理和逻辑上，每个节点或用户只有一个本地 MSP。但是，由于通道 MSP 对通道中的所有节点都可用，因此在通道配置中逻辑上只定义一次。但是，通道 MSP 也在通道中每个节点的文件系统上实例化，并通过协商一致保持同步。因此，尽管每个节点的本地文件系统上都有每个通道 MSP 的副本，但从逻辑上讲，通道 MSP 驻留在通道或网络上并由通道或网络维护。

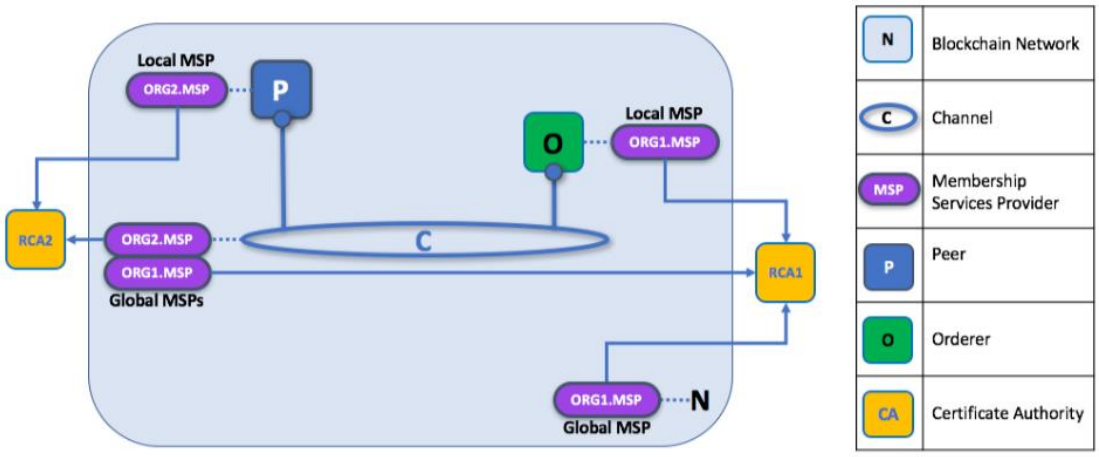
## 3、MSP Levels

## 3、MSP 水平

The split between channel and local MSPs reflects the needs of organizations to administer their local resources, such as a peer or orderer nodes, and their channel resources, such as ledgers, smart contracts, and consortia, which operate at the channel or network level. It’s helpful to think of these MSPs as being at different levels, with MSPs at a higher level relating to network administration concerns while MSPs at a lower level handle identity for the administration of private resources. MSPs are mandatory at every level of administration — they must be defined for the

network, channel, peer, orderer, and users.

渠道和本地 MSP 之间的划分反映了组织管理其本地资源（如对等或订购方节点）的需求，以及在渠道或网络级别运行的渠道资源（如分类账、智能合约和联合体）。将这些 MSP 视为处于不同级别是很有帮助的，较高级别的 MSP 与网络管理相关，而较低级别的 MSP 处理用于管理私有资源的标识。MSP 在每个管理级别都是强制的-必须为网络、通道、对等端、订购方和用户定义 MSP。




MSP Levels. The MSPs for the peer and orderer are local, whereas the MSPs for a channel (including the network configuration channel) are shared across all participants of that channel. In this figure, the network configuration channel is administered by ORG1, but another application channel can be managed by ORG1 and ORG2. The peer is a member of and managed by ORG2, whereas ORG1 manages the orderer of the figure. ORG1 trusts identities from RCA1, whereas ORG2 trusts identities from RCA2. Note that these are administration identities, reflecting who can administer these components. So while ORG1 administers the network, ORG2.MSP does exist in the network definition.

MSP 水平。对等端和订购方的 MSP 是本地的，而通道（包括网络配置通道）的 MSP 在该通道的所有参与者之间共享。在这个图中，网络配置通道由 org1 管理，但是另一个应用程序通道可以由 org1 和 org2 管理。对等机是 org2 的成员并由 org2 管理，而 org1 管理图的排序器。org1 信任来自 rca1 的标识，而 org2 信任来自 rca2 的标识。请注意，这些是管理标识，反映了谁可以管理这些组件。因此，当 org1 管理网络时，org2.msp 确实存在于网络定义中。

- **Network MSP:** The configuration of a network defines who are the members in the network — by defining the MSPs of the participant organizations — as well as which of these members are authorized to perform administrative tasks (e.g., creating a channel).
- **Channel MSP:** It is important for a channel to maintain the MSPs of its members separately. A channel provides private communications between a particular set of organizations which in turn have administrative control over it. Channel policies interpreted in the context of that channel’s MSPs define who has ability to participate in certain action

on the channel, e.g., adding organizations, or instantiating chaincodes. Note that there is no necessary relationship between the permission to administrate a channel and the ability to administrate the network configuration channel (or any other channel). Administrative rights exist within the scope of what is being administrated (unless the rules have been written otherwise — see the discussion of the `ROLE` attribute below).

- **Peer MSP:** This local MSP is defined on the file system of each peer and there is a single MSP instance for each peer. Conceptually, it performs exactly the same function as channel MSPs with the restriction that it only applies to the peer where it is defined. An example of an action whose authorization is evaluated using the peer's local MSP is the installation of a chaincode on the peer.
- **Orderer MSP:** Like a peer MSP, an orderer local MSP is also defined on the file system of the node and only applies to that node. Like peer nodes, orderers are also owned by a single organization and therefore have a single MSP to list the actors or nodes it trusts.
- **网络 MSP：**网络配置通过定义参与者组织的 MSP 来定义谁是网络中的成员，以及哪些成员被授权执行管理任务（例如，创建通道）。
- **通道 MSP：**频道必须单独维护其成员的 MSP。一个通道在一组特定的组织之间提供私有通信，而这些组织又对其具有管理控制权。在该通道的 MSP 上下文中解释的通道策略定义了谁有能力参与通道上的某些操作，例如添加组织或实例化链码。请注意，管理通道的权限和管理网络配置通道（或任何其他通道）的能力之间没有必要的关系。管理权存在于所管理的范围内（除非规则另有规定-请参阅下面的角色属性讨论）。
- **对等 MSP：**这个本地 MSP 是在每个对等机的文件系统上定义的，并且每个对等机只有一个 MSP 实例。从概念上讲，它执行与通道 MSP 完全相同的功能，但有一个限制，即它只适用于定义它的对等端。使用对等机的本地 MSP 评估其授权的操作的一个示例是在对等机上安装链码。
- **订购者 MSP：**与对等 MSP 一样，订购者本地 MSP 也在节点的文件系统上定义，并且仅应用于该节点。与对等节点一样，订购者也属于单个组织，因此拥有一个 MSP 来列出它信任的参与者或节点。

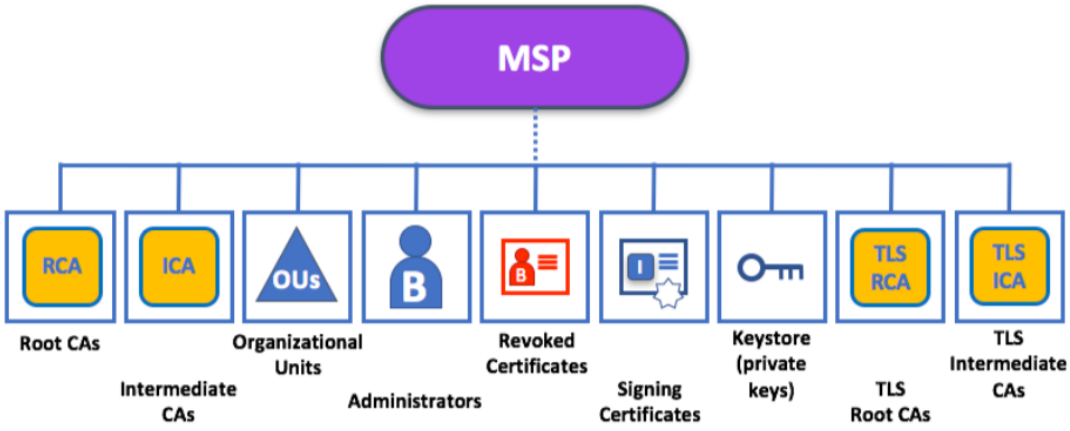
4、MSP Structure

4、MSP 结构

So far, you've seen that the most important element of an MSP are the specification of the root or intermediate CAs that are used to establish an actor's or node's membership in the respective organization. There are, however, more elements that are used in conjunction with these two to assist

with membership functions.

到目前为止，您已经看到了 MSP 最重要的元素是根 CA 或中间 CA 的规范，这些规范用于建立参与者或节点在各自组织中的成员资格。但是，还有更多的元素与这两个元素结合使用，以帮助成员功能。




The figure above shows how a local MSP is stored on a local filesystem. Even though channel MSPs are not physically structured in exactly this way, it's still a helpful way to think about them.

上图显示了本地 MSP 是如何存储在本地文件系统上的。尽管信道 MSP 的物理结构不是这样的，但它仍然是一种有帮助的思考方式。

As you can see, there are nine elements to an MSP. It's easiest to think of these elements in a directory structure, where the MSP name is the root folder name with each subfolder representing different elements of an MSP configuration.

如您所见，MSP 有九个元素。在目录结构中最容易想到这些元素，其中 msp 名称是根文件夹名称，每个子文件夹代表一个 msp 配置的不同元素。

Let's describe these folders in a little more detail and see why they are important.

让我们更详细地描述一下这些文件夹，看看它们为什么重要。

- **Root CAs:** This folder contains a list of self-signed X.509 certificates of the Root CAs trusted by the organization represented by this MSP. There must be at least one Root CA X.509 certificate in this MSP folder.

  This is the most important folder because it identifies the CAs from which all other certificates must be derived to be considered members of the corresponding organization.

- **Intermediate CAs:** This folder contains a list of X.509 certificates of the Intermediate CAs trusted by this organization. Each certificate must be signed by one of the Root CAs in the MSP or by an Intermediate CA whose issuing CA chain ultimately leads back to a trusted Root CA.

  An intermediate CA may represent a different subdivision of the organization (like `ORG1-MANUFACTURING` and `ORG1-DISTRIBUTION` do

for `ORG1`), or the organization itself (as may be the case if a commercial CA is leveraged for the organization's identity management). In the latter case intermediate CAs can be used to represent organization subdivisions. Here you may find more information on best practices for MSP configuration. Notice, that it is possible to have a functioning network that does not have an Intermediate CA, in which case this folder would be empty.

Like the Root CA folder, this folder defines the CAs from which certificates must be issued to be considered members of the organization.

- **Organizational Units (OUs):** These are listed in the `$FABRIC_CFG_PATH/msp/config.yaml` file and contain a list of organizational units, whose members are considered to be part of the organization represented by this MSP. This is particularly useful when you want to restrict the members of an organization to the ones holding an identity (signed by one of MSP designated CAs) with a specific OU in it.

  Specifying OUs is optional. If no OUs are listed, all the identities that are part of an MSP — as identified by the Root CA and Intermediate CA folders — will be considered members of the organization.
- **Administrators:** This folder contains a list of identities that define the actors who have the role of administrators for this organization. For the standard MSP type, there should be one or more X.509 certificates in this list.

  It's worth noting that just because an actor has the role of an administrator it doesn't mean that they can administer particular resources! The actual power a given identity has with respect to administering the system is determined by the policies that manage system resources. For example, a channel policy might specify that `ORG1-MANUFACTURING` administrators have the rights to add new organizations to the channel, whereas the `ORG1-DISTRIBUTION` administrators have no such rights.

Even though an X.509 certificate has a `ROLE` attribute (specifying, for example, that an actor is an `admin`), this refers to an actor's role within its organization rather than on the blockchain network. This is similar to the purpose of the `OU` attribute, which — if it has been defined — refers to an actor's place in the organization.

The `ROLE` attribute **can** be used to confer administrative rights at the channel level if the policy for that channel has been written to allow any administrator from an organization (or certain organizations) permission to perform certain channel functions (such as instantiating chaincode). In this way, an organizational role can confer a network role.

- **Revoked Certificates:** If the identity of an actor has been revoked, identifying information about the identity — not the identity itself — is held in this folder. For X.509-based identities, these identifiers are pairs of strings known as Subject Key Identifier (SKI) and Authority Access Identifier (AKI), and are checked whenever the X.509 certificate is being used to make sure the certificate has not been revoked.

  This list is conceptually the same as a CA's Certificate Revocation List (CRL), but it also relates to revocation of membership from the organization. As a result, the administrator of an MSP, local or channel, can quickly revoke an actor or node from an organization by advertising the updated CRL of the CA the revoked certificate as issued by. This "list of lists" is optional. It will only become populated as certificates are revoked.
- **Node Identity:** This folder contains the identity of the node, i.e., cryptographic material that — in combination to the content of `KeyStore` — would allow the node to authenticate itself in the messages that is sends to other participants of its channels and network. For X.509 based identities, this folder contains an **X.509 certificate**. This is the certificate a peer places in a transaction proposal response, for example, to indicate that the peer has endorsed it — which can subsequently be checked against the resulting transaction's endorsement policy at validation time.

This folder is mandatory for local MSPs, and there must be exactly one X.509 certificate for the node. It is not used for channel MSPs.

- KeyStore **for Private Key:** This folder is defined for the local MSP of a peer or orderer node (or in an client's local MSP), and contains the node's **signing key**. This key matches cryptographically the node's identity included in **Node Identity** folder and is used to sign data — for example to sign a transaction proposal response, as part of the endorsement phase.

  This folder is mandatory for local MSPs, and must contain exactly one private key. Obviously, access to this folder must be limited only to the identities of users who have administrative responsibility on the peer.

  Configuration of a **channel MSPs** does not include this folder, as channel MSPs solely aim to offer identity validation functionalities and not signing abilities.
- **TLS Root CA:** This folder contains a list of self-signed X.509 certificates of the Root CAs trusted by this organization **for TLS communications**. An example of a TLS communication would be when a peer needs to connect to an orderer so that it can receive ledger updates.

  MSP TLS information relates to the nodes inside the network — the peers and the orderers, in other words, rather than the applications and administrations that consume the network.

  There must be at least one TLS Root CA X.509 certificate in this folder.
- **TLS Intermediate CA:** This folder contains a list intermediate CA certificates CAs trusted by the organization represented by this MSP **for TLS communications**. This folder is specifically useful when commercial CAs are used for TLS certificates of an organization. Similar to membership intermediate CAs, specifying intermediate TLS CAs is optional.
- 根 CA:此文件夹包含此 MSP 表示的组织信任的根 CA 的自签名 X.509 证书列表。此 MSP 文件夹中必须至少有一个根 CA X.509 证书。

  这是最重要的文件夹，因为它标识必须从中派生所有其他证书才能被视为相应组织的成员的 CA。

- 中间 CA:此文件夹包含此组织信任的中间 CA 的 X.509 证书列表。每个证书必须由 MSP 中的一个根 CA 或中间 CA 签名，中间 CA 的颁发 CA 链最终将返回到受信任的根 CA。

  中间 CA 可以代表组织的不同分支（如 ORG1-Manufacturing 和 ORG1-Distribution Do for ORG1），也可以代表组织本身（如果商业 CA 用于组织的身份管理，则可能是这种情况）。在后一种情况下，中间 CA 可以用来表示组织的分支。您可以在这里找到有关 MSP 配置最佳实践的更多信息。请注意，可能有一个没有中间 CA 的正常工作的网络，在这种情况下，此文件夹将为空。

  与根 CA 文件夹一样，此文件夹定义了必须从中颁发证书才能被视为组织成员的 CA。

- 组织单位（OU）：这些单位列在$fabric_cfg_path/msp/config.yaml 文件中，并包含组织单位列表，其成员被视为该 msp 所代表组织的一部分。当您希望将组织的成员限制为拥有标识（由 MSP 指定的 CA 之一签名）且其中包含特定 OU 的成员时，这尤其有用。

  指定 OU 是可选的。如果没有列出 OU，那么属于 MSP 的所有标识（由根 CA 和中间 CA 文件夹标识）将被视为组织的成员。

- 管理员：此文件夹包含定义具有此组织管理员角色的参与者的标识列表。对于标准 MSP 类型，此列表中应该有一个或多个 X.509 证书。值得注意的是，仅仅因为参与者具有管理员的角色，并不意味着他们可以管理特定的资源！给定标识在管理系统方面的实际权限由管理系统资源的策略确定。例如，渠道策略可能指定 ORG1-Manufacturing 管理员有权向渠道添加新组织，而 ORG1-Distribution 管理员没有此类权限。

  尽管 X.509 证书具有角色属性（例如，指定参与者是管理员），但这指的是参与者在其组织内而不是在区块链网络上的角色。这类似于 ou 属性的用途，如果定义了它，那么它指的是参与者在组织中的位置。

  如果为某个组织（或某些组织）的任何管理员编写了允许其执行某些通道功能（例如实例化链码）的策略，则可以使用角色属性授予通道级别的管理权限。这样，组织角色可以授予网络角色。

- 吊销的证书：如果参与者的身份已被吊销，则有关该身份的标识信息（而不是标识本身）将保存在此文件夹中。对于基于 X.509 的标识，这些标识符是称为主题密钥标识符（SKI）和授权访问标识符（AKI）的字符串对，每当使用 X.509 证书时都会进行检查，以确保证书没有被吊销。

  此列表在概念上与 CA 的证书吊销列表（CRL）相同，但它也与组织成员资格的吊销有关。因此，MSP（本地或通道）的管理员可以通过将由颁发的已吊销证书作为 CA

的更新的 CRL 进行广告，从而快速从组织中吊销参与者或节点。此“列表”是可选的。它将仅在证书被吊销时填充。

- 节点标识：此文件夹包含节点的标识，即密码材料（与密钥库的内容结合使用），它允许节点在发送给其通道和网络的其他参与者的消息中进行身份验证。对于基于 X.509 的标识，此文件夹包含 X.509 证书。这是一个对等方在事务建议响应中放置的证书，例如，用于指示对等方已对其进行了批核-随后可以在验证时对照生成的事务的批核策略对其进行检查。

  此文件夹对于本地 MSP 是必需的，并且节点必须只有一个 X.509 证书。它不用于信道 MSP。

- 私钥的密钥库：此文件夹是为对等节点或订购方节点的本地 MSP（或在客户端的本地 MSP 中）定义的，并包含节点的签名密钥。此密钥以加密方式匹配节点标识文件夹中包含的节点标识，并用于对数据进行签名-例如，作为认可阶段的一部分，对事务建议响应进行签名。

  此文件夹对于本地 MSP 是必需的，并且必须正好包含一个私钥。显然，对该文件夹的访问必须仅限于对对等机负有管理责任的用户的标识。

  通道 MSP 的配置不包括此文件夹，因为通道 MSP 仅旨在提供身份验证功能，而不是签名功能。

- TLS 根 CA:此文件夹包含此组织信任的用于 TLS 通信的根 CA 的自签名 X.509 证书列表。TLS 通信的一个例子是，当对等方需要连接到订购方以便它可以接收分类帐更新时。

  msp tls 信息与网络内的节点相关，也就是说，对等端和订购方，而不是使用网络的应用程序和管理。

  此文件夹中必须至少有一个 TLS 根 CA X.509 证书。

- TLS 中间 CA:此文件夹包含一个列表，中间 CA 证书 CA 受此 MSP 代表的组织信任，用于进行 TLS 通信。当商业 CA 用于组织的 TLS 证书时，此文件夹特别有用。与成员身份中间 CA 类似，指定中间 TLS CA 是可选的。

  For more information about TLS, click here.
  有关 TLS 的详细信息，请单击此处。

If you’ve read this doc as well as our doc on Identity), you should have a pretty good grasp of how identities and membership work in Hyperledger Fabric. You’ve seen how a PKI and MSPs are used to identify the actors collaborating in a blockchain network. You’ve learned how certificates, public/private keys, and roots of trust work, in addition to how MSPs are physically and logically structured.

如果您已经阅读过本文档以及我们的身份文档，那么您应该很好地了解身份和成员

身份在 Hyperledger 结构中的工作方式。您已经看到了如何使用 pki 和 msp 识别区块链网络中协作的参与者。除了 MSP 的物理和逻辑结构之外，您还了解了证书、公钥/私钥和信任根是如何工作的。

# 七、Peers

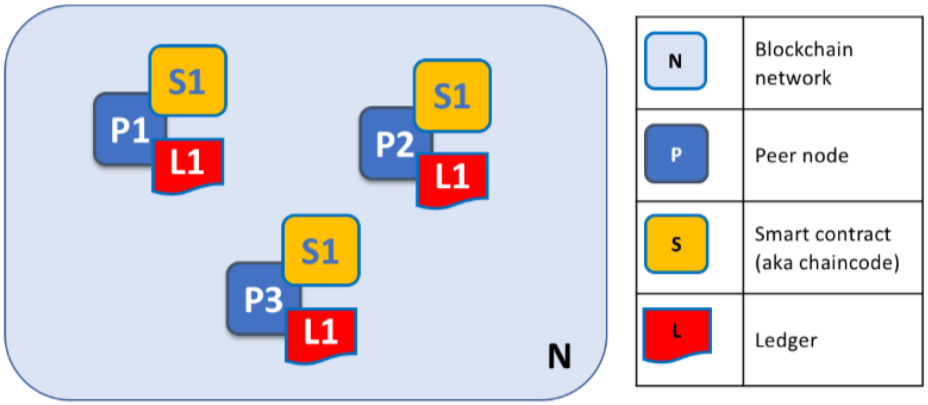
# 七、节点

The ordering of transactions is delegated to a modular component for consensus that is logically decoupled from the peers that execute transactions and maintain the ledger. Specifically, the ordering service. Since consensus is modular, its implementation can be tailored to the trust assumption of a particular deployment or solution. This modular architecture allows the platform to rely on well-established toolkits for CFT (crash fault-tolerant) or BFT (byzantine fault-tolerant) ordering. A blockchain network is comprised primarily of a set of peer nodes (or, simply, peers). Peers are a fundamental element of the network because they host ledgers and smart contracts. Recall that a ledger immutably records all the transactions generated by smart contracts (or chaincode). Smart contracts and ledgers are used to encapsulate the shared processes and shared information in a network, respectively. These aspects of a peer make them a good starting point to understand a Hyperledger Fabric network.

区块链网络主要由一组对等节点（或简单地说，对等节点）组成。节点是网络的一个基本要素，因为他们拥有分类账和智能合约。回想一下，分类账不可变地记录智能合约（或链码）生成的所有交易。智能合约和分类账分别用于封装网络中的共享流程和共享信息。同龄人的这些方面使他们成为了解超级账本结构网络的良好起点。

Other elements of the blockchain network are of course important: ledgers and smart contracts, orderers, policies, channels, applications, organizations, identities, and membership, and you can read more about them in their own dedicated sections. This section focusses on peers, and their relationship to those other elements in a Hyperledger Fabric network.

当然，区块链网络的其他要素也很重要：分类账和智能合约、订购者、政策、渠道、应用程序、组织、身份和会员资格，您可以在各自的专用章节中阅读更多关于它们的信息。本节主要讨论同行，以及他们与 Hyperledger 结构网络中其他元素的关系。



A blockchain network is comprised of peer nodes, each of which can hold copies of ledgers and copies of smart contracts. In this example, the network N consists of peers P1, P2 and P3, each of which maintain their own instance of the distributed ledger L1. P1, P2 and P3 use the same chaincode, S1, to access their copy of that distributed ledger.

区块链网络由对等节点组成，每个节点可以保存分类账副本和智能合约副本。在这个例子中，网络 n 由对等方 p1、p2 和 p3 组成，每个对等方维护自己的分布式账本 l1 实例。p1、p2 和 p3 使用相同的链码 s1 访问它们的分布式分类账副本。

Peers can be created, started, stopped, reconfigured, and even deleted. They expose a set of APIs that enable administrators and applications to interact with the services that they provide. We'll learn more about these services in this section.

可以创建、启动、停止、重新配置甚至删除对等点。它们公开了一组 API，使管理员和应用程序能够与它们提供的服务交互。我们将在本节中进一步了解这些服务。

## 1、A word on terminology

## 1、术语一词

Hyperledger Fabric implements smart contracts with a technology concept it calls chaincode — simply a piece of code that accesses the ledger, written in one of the supported programming languages. In this topic, we'll usually use the term chaincode, but feel free to read it as smart contract if you're more used to that term. It's the same thing!

HyperledgeFabric 实现了智能合约，其技术概念称为 chaincode——只是一段访问分类账的代码，用一种支持的编程语言编写。在本主题中，我们通常会使用术语 chaincode，但如果您更习惯使用该术语，可以将其作为智能合约阅读。这是同一件事！

## 2、Ledgers and Chaincode

## 2、分类帐和链码

Let's look at a peer in a little more detail. We can see that it's the peer that hosts both the ledger and chaincode. More accurately, the peer actually hosts instances of the ledger, and instances of chaincode. Note that this provides a deliberate redundancy in a Fabric network — it avoids single points of failure. We'll learn more about the distributed and decentralized nature of a blockchain network later in this section.

让我们更详细地看一看同行。我们可以看到，正是同龄人承载了分类账和链码。更准确地说，对等机实际上承载了分类帐实例和链代码实例。注意，这在结构网络中提供了一个有意的冗余——它避免了单点故障。我们将在本节稍后了解区块链网络的分布式和分散性。


A peer hosts instances of ledgers and instances of chaincodes. In this example, P1 hosts an instance of ledger L1 and an instance of chaincode S1. There can be many ledgers and chaincodes hosted on an individual peer.

对等机承载分类帐实例和链码实例。在本例中，p1 承载分类帐 l1 的实例和链代码 s1 的实例。可以有许多分类帐和链码托管在单个对等机上。

Because a peer is a host for ledgers and chaincodes, applications and administrators must interact with a peer if they want to access these resources. That's why peers are considered the most fundamental building blocks of a Hyperledger Fabric network. When a peer is first created, it has neither ledgers nor chaincodes. We'll see later how ledgers get created, and how chaincodes get installed, on peers.

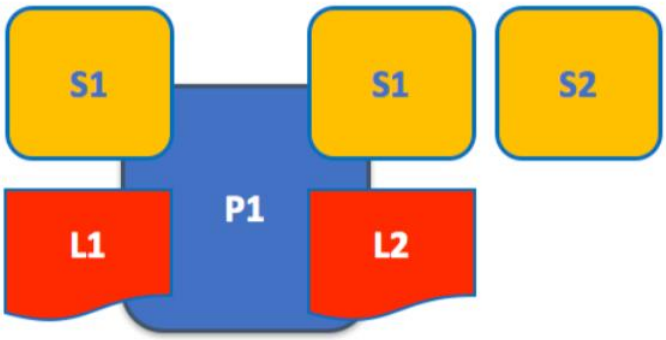
因为对等机是分类帐和链码的主机，所以如果应用程序和管理员想要访问这些资源，他们必须与对等机进行交互。这就是为什么同行被认为是超级账本结构网络最基本的组成部分。第一次创建对等机时，它既没有分类帐，也没有链码。稍后我们将看到如何创建分类帐，以及如何在对等机上安装链码。

**Multiple Ledgers**

**多个分类帐**

A peer is able to host more than one ledger, which is helpful because it allows for a flexible system design. The simplest configuration is for a peer to manage a single ledger, but it's absolutely appropriate for a peer to host two or more ledgers when required.

一个对等方能够承载多个分类账，这很有帮助，因为它允许灵活的系统设计。最简单的配置是让一个对等方管理一个分类账，但是当需要时，它绝对适合一个对等方托管两个或更多分类账。




A peer hosting multiple ledgers. Peers host one or more ledgers, and each ledger has zero or more chaincodes that apply to them. In this example, we can see that the peer P1 hosts ledgers L1 and L2. Ledger L1 is accessed using chaincode S1. Ledger L2 on the other hand can be accessed using chaincodes S1 and S2.

承载多个分类帐的对等机。对等机承载一个或多个分类帐，并且每个分类帐都有零个或多个适用于它们的链码。在这个例子中，我们可以看到对等机 p1 承载分类帐 l1 和 l2。使用链码 s1 访问分类帐 l1。另一方面，可以使用链代码 s1 和 s2 访问分类帐 l2。

Although it is perfectly possible for a peer to host a ledger instance without hosting any chaincodes which access that ledger, it’s rare that peers are configured this way. The vast majority of peers will have at least one chaincode installed on it which can query or update the peer’s ledger instances. It’s worth mentioning in passing that, whether or not users have installed chaincodes for use by external applications, peers also have special system chaincodes that are always present. These are not discussed in detail in this topic.

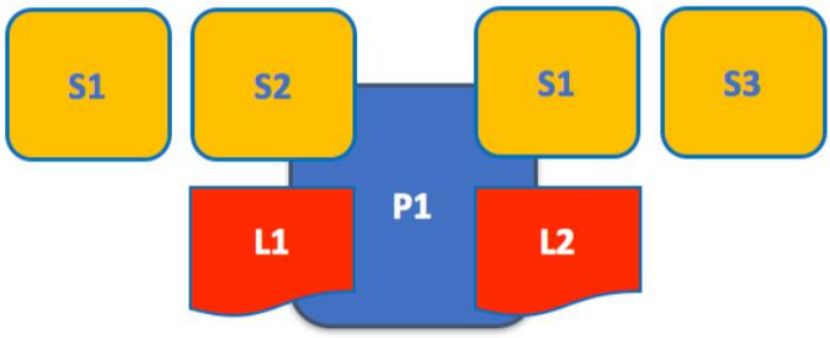
尽管对等端完全可以在不托管任何访问该分类帐的链码的情况下托管分类帐实例，但很少有对等端以这种方式配置。绝大多数对等方将至少安装一个可以查询或更新对等方分类账实例的链码。值得一提的是，无论用户是否已经安装了供外部应用程序使用的链码，对等端也有一直存在的特殊系统链码。本主题不详细讨论这些问题。

**Multiple Chaincodes**

**多链码**

There isn’t a fixed relationship between the number of ledgers a peer has and the number of chaincodes that can access that ledger. A peer might have many chaincodes and many ledgers available to it.

对等方拥有的分类账数量和可以访问该分类账的链码数量之间没有固定的关系。一个对等机可能有许多链码和许多可用的分类帐。




An example of a peer hosting multiple chaincodes. Each ledger can have many chaincodes which access it. In this example, we can see that peer P1 hosts ledgers L1 and L2, where L1 is accessed by chaincodes S1 and S2, and L2 is accessed by S1 and S3. We can see that S1 can access both L1 and L2.

承载多个链码的对等机示例。每个分类账可以有许多链码来访问它。在这个例子中，我们可以看到对等端 p1 承载着分类帐 l1 和 l2，其中 l1 由链码 s1 和 s2 访问，l2 由 s1 和 s3 访问。我们可以看到 s1 可以同时访问 l1 和 l2。

We’ ll see a little later why the concept of channels in Hyperledger Fabric is important when hosting multiple ledgers or multiple chaincodes on a peer.

稍后我们将看到为什么在一个对等端承载多个分类账或多个链码时，Hyperledger 结构中的通道概念很重要。

## 3、Applications and Peers

## 3、应用程序和对等机

We’ re now going to show how applications interact with peers to access the ledger. Ledger-query interactions involve a simple three-step dialogue between an application and a peer; ledger-update interactions are a little more involved, and require two extra steps. We’ ve simplified these steps a little to help you get started with Hyperledger Fabric, but don’ t worry — what’ s most important to understand is the difference in application-peer interactions for ledger-query compared to ledger-update transaction styles.

我们现在将展示应用程序如何与同行交互以访问分类账。分类帐查询交互涉及应用程序和对等机之间的简单的三步对话；分类帐更新交互稍微复杂一些，需要两个额外的步骤。我们稍微简化了这些步骤，以帮助您开始使用 Hyperledger 结构，但不要担心-最重要的是要了解的是，与分类帐更新交易样式相比，分类帐查询的应用程序对等交互之间的差异。

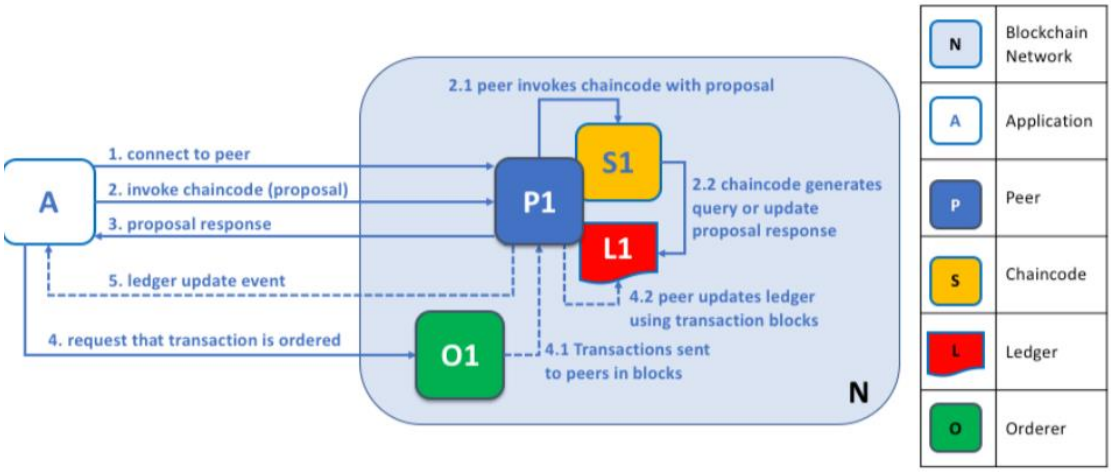
Applications always connect to peers when they need to access ledgers and chaincodes. The Hyperledger Fabric Software Development Kit (SDK) makes this easy for programmers — its APIs enable applications to connect to peers, invoke chaincodes to generate transactions, submit transactions to the

network that will get ordered and committed to the distributed ledger, and receive events when this process is complete.

当需要访问分类帐和链码时，应用程序总是连接到对等机。Hyperledger Fabric 软件开发工具包（SDK）使程序员更容易实现这一点——它的 API 使应用程序能够连接到对等端，调用链码来生成事务，将事务提交到网络，这些事务将被订购并提交到分布式账本，并在该过程完成时接收事件。

Through a peer connection, applications can execute chaincodes to query or update a ledger. The result of a ledger query transaction is returned immediately, whereas ledger updates involve a more complex interaction between applications, peers and orderers. Let's investigate this in a little more detail.

通过对等连接，应用程序可以执行链码来查询或更新分类帐。分类帐查询事务的结果将立即返回，而分类帐更新涉及应用程序、对等方和订购方之间更复杂的交互。让我们更详细地研究一下。




Peers, in conjunction with orderers, ensure that the ledger is kept up-to-date on every peer. In this example, application A connects to P1 and invokes chaincode S1 to query or update the ledger L1. P1 invokes S1 to generate a proposal response that contains a query result or a proposed ledger update. Application A receives the proposal response and, for queries, the process is now complete. For updates, A builds a transaction from all of the responses, which it sends it to O1 for ordering. O1 collects transactions from across the network into blocks, and distributes these to all peers, including P1. P1 validates the transaction before applying to L1. Once L1 is updated, P1 generates an event, received by A, to signify completion.

同行与订购方一起确保每一个同行的账本都是最新的。在本例中，应用程序 A 连接到 p1 并调用链代码 s1 以查询或更新分类帐 l1。p1 调用 s1 以生成包含查询结果或建议的分类帐更新的建议响应。应用程序 A 接收提案响应，对于查询，该过程现在已完成。对于更新，A 从所有响应构建一个交易，并将其发送给 O1 进行排序。O1 将跨网络的事务收集到块中，并将这些交易分发给所有对等方，包括 P1。p1 在应用于 l1 之前验证事务。一旦 l1 被更新，p1 生成一个交易，由 a 接收，表示完成。

A peer can return the results of a query to an application immediately since all of the information required to satisfy the query is in the peer's

local copy of the ledger. Peers never consult with other peers in order to respond to a query from an application. Applications can, however, connect to one or more peers to issue a query; for example, to corroborate a result between multiple peers, or retrieve a more up-to-date result from a different peer if there's a suspicion that information might be out of date. In the diagram, you can see that ledger query is a simple three-step process.

对等方可以立即将查询结果返回到应用程序，因为满足查询所需的所有信息都在对等方的分类帐本地副本中。对等方从不与其他对等方协商以响应来自应用程序的查询。但是，应用程序可以连接到一个或多个对等方来发出查询；例如，如果怀疑信息可能已过时，可以在多个对等方之间证实结果，或者从其他对等方检索更新的结果。在图中，您可以看到分类帐查询是一个简单的三步过程。

An update transaction starts in the same way as a query transaction, but has two extra steps. Although ledger-updating applications also connect to peers to invoke a chaincode, unlike with ledger-querying applications, an individual peer cannot perform a ledger update at this time, because other peers must first agree to the change — a process called consensus. Therefore, peers return to the application a proposed update — one that this peer would apply subject to other peers' prior agreement. The first extra step — step four — requires that applications send an appropriate set of matching proposed updates to the entire network of peers as a transaction for commitment to their respective ledgers. This is achieved by the application using an orderer to package transactions into blocks, and distribute them to the entire network of peers, where they can be verified before being applied to each peer's local copy of the ledger. As this whole ordering processing takes some time to complete (seconds), the application is notified asynchronously, as shown in step five.

更新交易的启动方式与查询交易相同，但有两个额外步骤。尽管分类帐更新应用程序也连接到对等方以调用链码，但与分类帐查询应用程序不同，单个对等方此时无法执行分类帐更新，因为其他对等方必须首先同意更改，这一过程称为共识。因此，对等方返回给应用程序一个建议的更新-此对等方将根据其他对等方的事先协议应用此更新。第一个额外的步骤-第四步-要求应用程序向整个对等网络发送一组适当的匹配建议更新，作为对各自分类账的承诺交易。这是通过应用程序使用排序将事务打包成块，然后将它们分发到整个对等网络，在将它们应用到每个对等的分类账的本地副本之前，可以在该网络中对它们进行验证。由于整个排序处理需要一些时间才能完成（秒），因此将异步通知应用程序，如步骤 5 所示。

Later in this section, you'll learn more about the detailed nature of this ordering process — and for a really detailed look at this process see the Transaction Flow topic.

在本节的后面部分，您将了解有关此订购流程的详细性质的更多信息—有关此流程的真正详细信息，请参阅事务流主题。

## 4、Peers and Channels

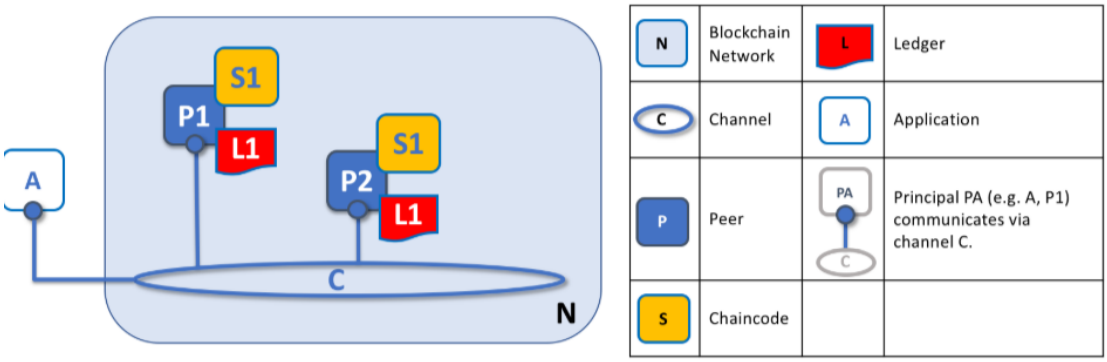
## 4、对等节点和通道

Although this section is about peers rather than channels, it's worth spending a little time understanding how peers interact with each other, and with applications, via channels — a mechanism by which a set of components within a blockchain network can communicate and transact privately.

尽管这一部分是关于对等方而非通道的，但值得花点时间了解对等方如何通过渠道相互作用以及与应用程序的相互作用——一种机制，通过这种机制，区块链网络中的一组组件可以进行通信和私下交易。

These components are typically peer nodes, orderer nodes and applications and, by joining a channel, they agree to collaborate to collectively share and manage identical copies of the ledger associated with that channel. Conceptually, you can think of channels as being similar to groups of friends (though the members of a channel certainly don't need to be friends!). A person might have several groups of friends, with each group having activities they do together. These groups might be totally separate (a group of work friends as compared to a group of hobby friends), or there can be some crossover between them. Nevertheless, each group is its own entity, with "rules" of a kind.

这些组件通常是对等节点、订购方节点和应用程序，通过加入一个通道，它们同意协作，共同共享和管理与该通道关联的分类账的相同副本。从概念上讲，你可以认为频道与朋友群相似（尽管频道的成员当然不需要成为朋友！）一个人可能有几个朋友组，每个朋友组都有他们一起做的活动。这些群体可能是完全独立的（一组工作朋友与一组爱好朋友相比），或者他们之间可能有一些交叉点。然而，每个团体都是自己的实体，有着某种"规则"。




Channels allow a specific set of peers and applications to communicate with each other within a blockchain network. In this example, application A can communicate directly with peers P1 and P2 using channel C. You can think of the channel as a pathway for communications between particular applications and peers. (For simplicity, orderers are not shown in this diagram, but must be present in a functioning network.)

通道允许一组特定的对等点和应用程序在区块链网络内相互通信。在本例中，应用程序 A 可以使用通道 C 直接与对等端 P1 和 P2 通信。您可以将通道视为特定应用程序和对等端之间通信的路径。（为简单起见，此图中不显示订购者，但订购者必须存在于正常工作的网络中。）

We see that channels don't exist in the same way that peers do — it's

more appropriate to think of a channel as a logical structure that is formed by a collection of physical peers. It is vital to understand this point — peers provide the control point for access to, and management of, channels.

我们看到，通道的存在方式不同于对等端——将通道视为由物理对等端集合形成的逻辑结构更为恰当。理解这一点是至关重要的——同行提供了访问和管理渠道的控制点。

## 5、Peers and Organizations

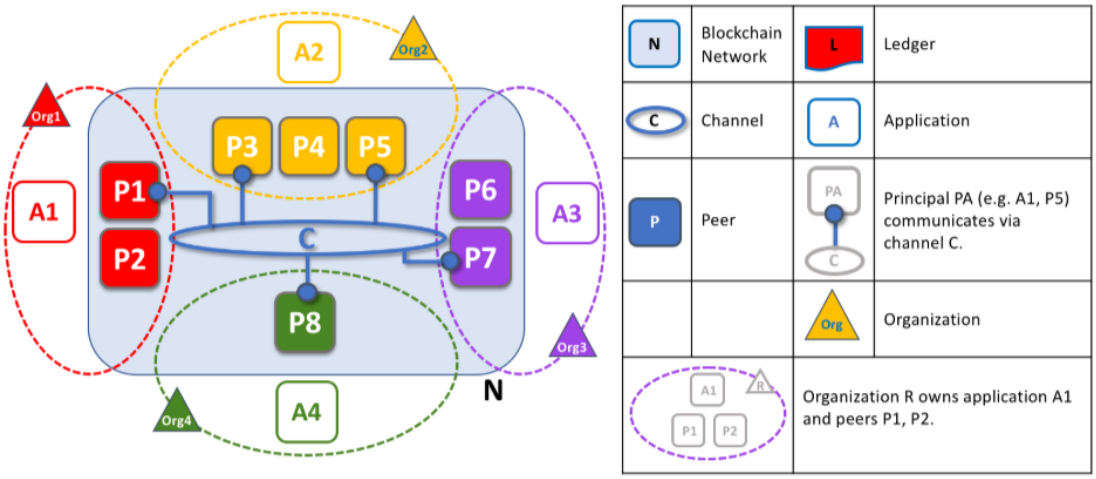
## 5、同行和组织

Now that you understand peers and their relationship to ledgers, chaincodes and channels, you'll be able to see how multiple organizations come together to form a blockchain network.

现在，您了解了对等点及其与分类账、链码和渠道的关系，您将能够看到多个组织如何聚集在一起形成一个区块链网络。

Blockchain networks are administered by a collection of organizations rather than a single organization. Peers are central to how this kind of distributed network is built because they are owned by — and are the connection points to the network for — these organizations.

区块链网络由一组组织而不是单个组织管理。对等点对于这种分布式网络的构建至关重要，因为它们属于这些组织，并且是这些组织的网络连接点。




Peers in a blockchain network with multiple organizations. The blockchain network is built up from the peers owned and contributed by the different organizations. In this example, we see four organizations contributing eight peers to form a network. The channel C connects five of these peers in the network N — P1, P3, P5, P7 and P8. The other peers owned by these organizations have not been joined to this channel, but are typically joined to at least one other channel. Applications that have been developed by a particular organization will connect to their own organization's peers as well as those of different organizations. Again, for simplicity, an orderer node is not shown in this diagram.

区块链网络中具有多个组织的同行。区块链网络是由不同组织拥有和贡献的同行建立的。在这个例子中，我们看到四个组织贡献八个对等方来形成一个网络。通道 C 连接

网络 n-p1、p3、p5、p7 和 p8 中的五个对等点。这些组织拥有的其他对等方尚未加入此通道，但通常至少加入一个其他通道。由一个特定组织开发的应用程序将连接到他们自己组织的对等方以及不同组织的对等方。同样，为了简单起见，此图中不显示医嘱者节点。

It’s really important that you can see what’s happening in the formation of a blockchain network. The network is both formed and managed by the multiple organizations who contribute resources to it. Peers are the resources that we’re discussing in this topic, but the resources an organization provides are more than just peers. There’s a principle at work here — the network literally does not exist without organizations contributing their individual resources to the collective network. Moreover, the network grows and shrinks with the resources that are provided by these collaborating organizations.

你能看到区块链网络的形成过程中发生了什么，这一点非常重要。网络由多个向其提供资源的组织组成和管理。同行是我们在本主题中讨论的资源，但组织提供的资源不仅仅是同行。这里有一个原则在起作用——如果没有组织将他们的个人资源贡献给集体网络，网络就根本不存在。此外，网络随着这些协作组织提供的资源而增长和缩小。

You can see that (other than the ordering service) there are no centralized resources — in the example above, the network, N, would not exist if the organizations did not contribute their peers. This reflects the fact that the network does not exist in any meaningful sense unless and until organizations contribute the resources that form it. Moreover, the network does not depend on any individual organization — it will continue to exist as long as one organization remains, no matter which other organizations may come and go. This is at the heart of what it means for a network to be decentralized.

您可以看到（除了订购服务之外）没有集中的资源-在上面的示例中，如果组织没有贡献他们的同行，那么网络 n 将不存在。这反映了这样一个事实，即网络在任何意义上都不存在，除非和直到组织贡献构成网络的资源。此外，网络并不依赖于任何一个组织——只要一个组织存在，它就会继续存在，不管其他组织可能来来去去去。这是网络去中心化的核心。

Applications in different organizations, as in the example above, may or may not be the same. That’s because it’s entirely up to an organization as to how its applications process their peers’ copies of the ledger. This means that both application and presentation logic may vary from organization to organization even though their respective peers host exactly the same ledger data.

不同组织中的应用程序，如上面的示例，可能相同，也可能不同。这是因为它完全取决于一个组织如何处理他们的同行的分类账副本。这意味着应用程序和表示逻辑可能因组织而异，即使它们各自的对等方托管着完全相同的分类帐数据。

Applications connect either to peers in their organization, or peers in another organization, depending on the nature of the ledger interaction that’s required. For ledger-query interactions, applications typically connect to their own organization’s peers. For ledger-update interactions,

we’ll see later why applications need to connect to peers representing every organization that is required to endorse the ledger update.

根据所需的分类帐交互的性质，应用程序要么连接到其组织中的对等方，要么连接到其他组织中的对等方。对于分类帐查询交互，应用程序通常连接到自己组织的对等方。对于分类帐更新交互，我们稍后将了解为什么应用程序需要连接到代表每个组织的对等方，这些组织需要对分类帐更新进行认可。

## 6、Peers and Identity

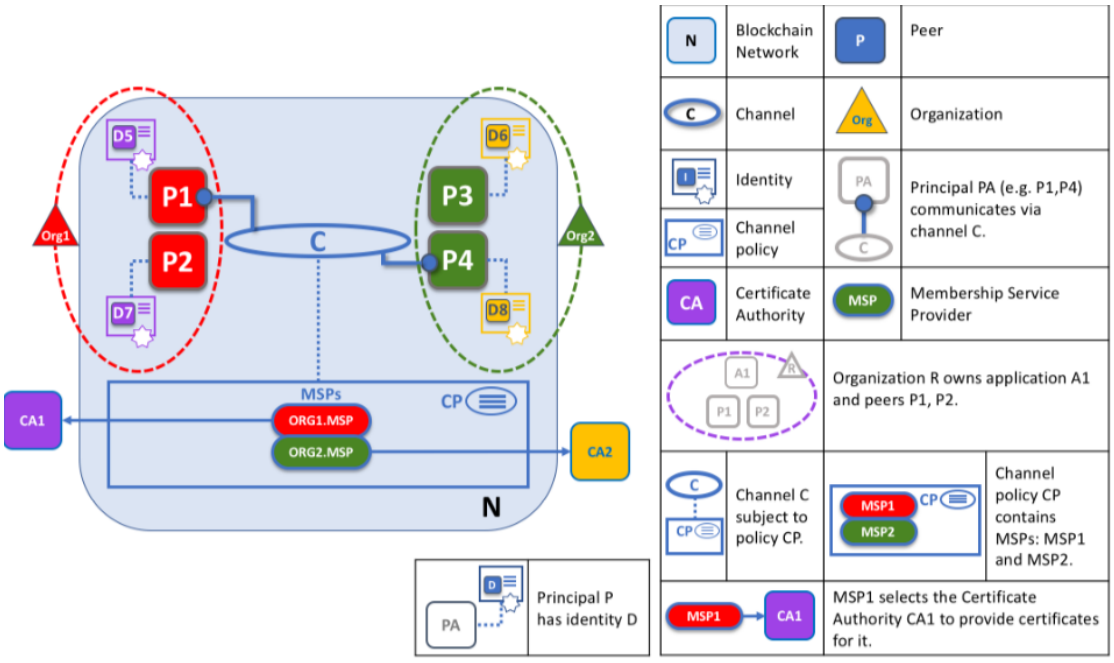
## 6、同行和身份

Now that you’ve seen how peers from different organizations come together to form a blockchain network, it’s worth spending a few moments understanding how peers get assigned to organizations by their administrators.

现在，您已经了解了不同组织的同行是如何聚集在一起形成区块链网络的，因此值得花一些时间了解同行是如何由其管理员分配给组织的。

Peers have an identity assigned to them via a digital certificate from a particular certificate authority. You can read lots more about how X.509 digital certificates work elsewhere in this guide but, for now, think of a digital certificate as being like an ID card that provides lots of verifiable information about a peer. Each and every peer in the network is assigned a digital certificate by an administrator from its owning organization.

对等方具有通过特定证书颁发机构的数字证书分配给他们的标识。在本指南的其他地方，您可以阅读更多关于 X.509 数字证书如何工作的信息，但现在，将数字证书视为提供大量可验证的对等信息的身份证。网络中的每个对等点都由其所属组织的管理员分配一个数字证书。




When a peer connects to a channel, its digital certificate identifies its owning organization via a channel MSP. In this example, P1 and P2 have identities issued by CA1. Channel C determines from a policy in its channel

configuration that identities from CA1 should be associated with Org1 using ORG1.MSP. Similarly, P3 and P4 are identified by ORG2.MSP as being part of Org2.

当对等端连接到通道时，其数字证书通过通道 MSP 标识其所属组织。在本例中，p1 和 p2 具有由 ca1 颁发的标识。通道 C 根据其通道配置中的策略确定，来自 CA1 的标识应该使用 org1.msp 与 org1 关联。同样，p3 和 p4 由 org2.msp 标识为 org2 的一部分。

Whenever a peer connects using a channel to a blockchain network, a policy in the channel configuration uses the peer's identity to determine its rights. The mapping of identity to organization is provided by a component called a Membership Service Provider (MSP) — it determines how a peer gets assigned to a specific role in a particular organization and accordingly gains appropriate access to blockchain resources. Moreover, a peer can be owned only by a single organization, and is therefore associated with a single MSP. We'll learn more about peer access control later in this section, and there's an entire section on MSPs and access control policies elsewhere in this guide. But for now, think of an MSP as providing linkage between an individual identity and a particular organizational role in a blockchain network.

当一个对等方使用一个通道连接到一个区块链网络时，通道配置中的策略使用对等方的身份来确定其权利。身份到组织的映射由一个称为会员服务提供商（MSP）的组件提供——它确定对等方如何分配到特定组织中的特定角色，并相应地获得对区块链资源的适当访问。此外，对等机只能由单个组织拥有，因此与单个 MSP 关联。我们将在本节后面了解更多关于对等访问控制的信息，本指南的其他部分还有一个完整的关于 MSP 和访问控制策略的部分。但就目前而言，将 MSP 视为在区块链网络中提供个人身份和特定组织角色之间的链接。

To digress for a moment, peers as well as everything that interacts with a blockchain network acquire their organizational identity from their digital certificate and an MSP. Peers, applications, end users, administrators and orderers must have an identity and an associated MSP if they want to interact with a blockchain network. We give a name to every entity that interacts with a blockchain network using an identity — a principal. You can learn lots more about principals and organizations elsewhere in this guide, but for now you know more than enough to continue your understanding of peers!

为了暂时摆脱困境，同龄人以及与区块链网络交互的所有人都从他们的数字证书和 MSP 中获取他们的组织身份。对等方、应用程序、最终用户、管理员和订购者如果想与区块链网络交互，必须具有标识和关联的 MSP。我们给每个使用身份（主体）与区块链网络交互的实体命名。在本指南的其他地方，您可以了解更多有关校长和组织的信息，但现在您已经知道了足够多的信息，可以继续了解同事了！

Finally, note that it's not really important where the peer is physically located — it could reside in the cloud, or in a data centre owned by one of the organizations, or on a local machine — it's the identity associated with it that identifies it as being owned by a particular organization. In our example above, P3 could be hosted in Org1's data center, but as long as the digital certificate associated with it is issued by CA2, then it's owned

by Org2.

最后，请注意，对等机的物理位置并不重要——它可能位于云上、某个组织拥有的数据中心或本地计算机上——与之相关联的标识将其标识为特定组织拥有。在上面的例子中,p3 可以托管在 org1 的数据中心,但是只要与之相关的数字证书是由 ca2 颁发的，那么它就属于 org2。

7、Peers and Orderers

7、对等方和订购方

We’ve seen that peers form the basis for a blockchain network, hosting ledgers and chaincode which can be queried and updated by peer-connected applications. However, the mechanism by which applications and peers interact with each other to ensure that every peer’s ledger is kept consistent is mediated by special nodes called orderers, and it’s to these nodes we now turn our attention.

我们已经看到，对等方构成了区块链网络的基础，托管分类账和链码，这些分类账和链码可以被对等连接的应用程序查询和更新。然而，应用程序和对等机相互作用以确保每个对等机的分类帐保持一致的机制是由称为“订购者”的特殊节点介导的，现在我们将注意力转向这些节点。

An update transaction is quite different from a query transaction because a single peer cannot, on its own, update the ledger — updating requires the consent of other peers in the network. A peer requires other peers in the network to approve a ledger update before it can be applied to a peer’s local ledger. This process is called consensus, which takes much longer to complete than a simple query. But when all the peers required to approve the transaction do so, and the transaction is committed to the ledger, peers will notify their connected applications that the ledger has been updated. You’re about to be shown a lot more detail about how peers and orderers manage the consensus process in this section.

更新事务与查询事务有很大的不同,因为单个对等方本身无法更新分类帐-更新需要网络中其他对等方的同意。对等方要求网络中的其他对等方批准分类帐更新，然后才能将其应用到对等方的本地分类帐。这个过程称为协商一致，这比简单的查询需要更长的时间才能完成。但是，当批准交易所需的所有对等方都这样做，并且交易提交到分类账时，对等方将通知其关联的应用程序分类账已更新。在本节中，您将看到更多关于同行和订购者如何管理协商一致过程的详细信息。

Specifically, applications that want to update the ledger are involved in a 3-phase process, which ensures that all the peers in a blockchain network keep their ledgers consistent with each other. In the first phase, applications work with a subset of endorsing peers, each of which provide an endorsement of the proposed ledger update to the application, but do not apply the proposed update to their copy of the ledger. In the second phase, these separate endorsements are collected together as transactions and packaged into blocks. In the final phase, these blocks are distributed back to every peer where each transaction is validated before being applied to that peer’s copy of the ledger.

具体来说，想要更新分类账的应用程序涉及到一个 3 阶段的过程，这可以确保区块

链网络中的所有对等方保持其分类账彼此一致。在第一阶段，应用程序与背书对等方的子集一起工作，每个对等方都向应用程序提供拟议分类帐更新的背书，但不将拟议更新应用于其分类帐副本。在第二阶段，这些单独的背书作为交易收集在一起，并打包成块。在最后一个阶段，这些块被分发回每一个对等方，每个交易在被应用到该对等方的分类账副本之前都经过了验证。

As you will see, orderer nodes are central to this process, so let's investigate in a little more detail how applications and peers use orderers to generate ledger updates that can be consistently applied to a distributed, replicated ledger.

正如您将看到的，订购方节点是这个过程的核心，所以让我们更详细地研究一下应用程序和对等方如何使用订购方来生成分类帐更新，这些更新可以一致地应用于分布式、复制的分类帐。

**Phase 1: Proposal**

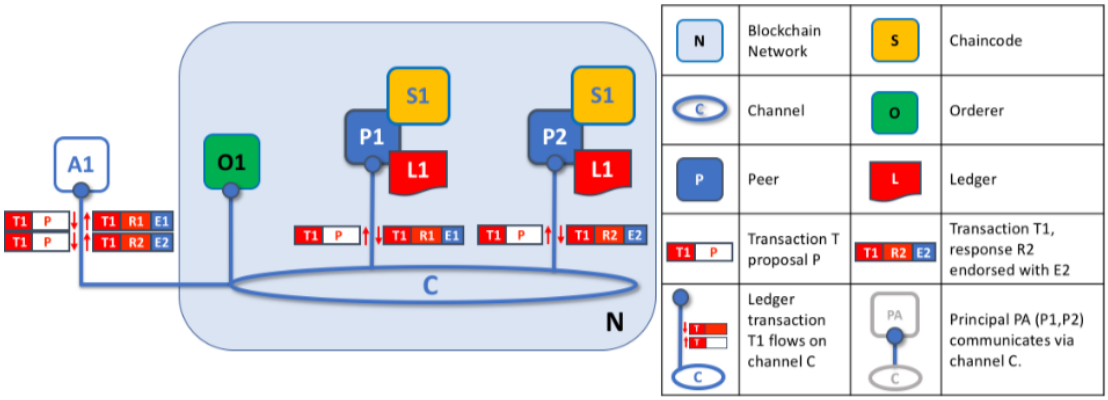
**第一阶段：提案**

Phase 1 of the transaction workflow involves an interaction between an application and a set of peers — it does not involve orderers. Phase 1 is only concerned with an application asking different organizations' endorsing peers to agree to the results of the proposed chaincode invocation.

交易工作流的第一阶段涉及应用程序和一组对等方之间的交互-它不涉及订购方。阶段 1 只涉及一个应用程序，要求不同组织的认可对等方同意提议的链代码调用的结果。

To start phase 1, applications generate a transaction proposal which they send to each of the required set of peers for endorsement. Each of these endorsing peers then independently executes a chaincode using the transaction proposal to generate a transaction proposal response. It does not apply this update to the ledger, but rather simply signs it and returns it to the application. Once the application has received a sufficient number of signed proposal responses, the first phase of the transaction flow is complete. Let's examine this phase in a little more detail.

为了开始阶段 1，应用程序生成一个交易提案，并将其发送给所需的每一组对等方进行认可。然后，这些认可对等体中的每一个使用交易建议独立地执行一个链代码，以生成交易建议响应。它不将此更新应用于分类帐，而是简单地签名并将其返回给应用程序。一旦应用程序收到足够数量的已签名的建议响应，交易流的第一阶段就完成了。让我们更详细地研究一下这个阶段。




Transaction proposals are independently executed by peers who return

endorsed proposal responses. In this example, application A1 generates transaction T1 proposal P which it sends to both peer P1 and peer P2 on channel C. P1 executes S1 using transaction T1 proposal P generating transaction T1 response R1 which it endorses with E1. Independently, P2 executes S1 using transaction T1 proposal P generating transaction T1 response R2 which it endorses with E2. Application A1 receives two endorsed responses for transaction T1, namely E1 and E2.

交易建议由返回认可的建议响应的同行独立执行。在此示例中，应用程序 A1 生成事务 T1 Proposal P，并将其发送到通道 C 上的对等端 P1 和对等端 P2。P1 使用事务 T1 Proposal P 执行 S1，生成事务 T1 响应 R1，并将其与 e1 背书。独立地，p2 使用事务 T1 Proposal P 执行 s1，生成事务 T1 响应 r2，并与 e2 背书。应用程序 A1 收到事务 T1 的两个认可响应，即 e1 和 e2。

Initially, a set of peers are chosen by the application to generate a set of proposed ledger updates. Which peers are chosen by the application? Well, that depends on the endorsement policy (defined for a chaincode), which defines the set of organizations that need to endorse a proposed ledger change before it can be accepted by the network. This is literally what it means to achieve consensus — every organization who matters must have endorsed the proposed ledger change before it will be accepted onto any peer's ledger.

最初，应用程序选择一组对等方来生成一组建议的分类帐更新。应用程序选择哪些对等机？嗯，这取决于认可政策（为一个链码定义），它定义了在被网络接受之前需要认可一个提议的分类账变更的一组组织。这正是达成共识的真正意义所在——每一个重要的组织都必须在提议的分类账变更被任何同行的分类账接受之前对其进行背书。

A peer endorses a proposal response by adding its digital signature, and signing the entire payload using its private key. This endorsement can be subsequently used to prove that this organization's peer generated a particular response. In our example, if peer P1 is owned by organization Org1, endorsement E1 corresponds to a digital proof that "Transaction T1 response R1 on ledger L1 has been provided by Org1's peer P1!".

对等方通过添加其数字签名和使用其私钥对整个有效负载进行签名来认可提案响应。这种认可可以随后用来证明这个组织的对等方生成了一个特定的响应。在我们的示例中，如果对等机 p1 归组织 org1 所有，那么背书 e1 对应于"分类帐 l1 上的事务 t1 响应 r1 已由组织的对等机 p1 提供的数字证明！"。

Phase 1 ends when the application receives signed proposal responses from sufficient peers. We note that different peers can return different and therefore inconsistent transaction responses to the application for the same transaction proposal. It might simply be that the result was generated at different times on different peers with ledgers at different states, in which case an application can simply request a more up-to-date proposal response. Less likely, but much more seriously, results might be different because the chaincode is non-deterministic. Non-determinism is the enemy of chaincodes and ledgers and if it occurs it indicates a serious problem with the proposed transaction, as inconsistent results cannot, obviously, be applied to ledgers.

An individual peer cannot know that their transaction result is non-deterministic — transaction responses must be gathered together for comparison before non-determinism can be detected. (Strictly speaking, even this is not enough, but we defer this discussion to the transaction section, where non-determinism is discussed in detail.)

当应用程序从足够的对等方收到签名的建议响应时，阶段 1 结束。我们注意到，不同的对等方可以返回不同的、因此不一致的对同一交易建议的应用程序的事务响应。它可能只是在不同的时间在不同的同龄人上生成了不同状态的分类账结果，在这种情况下，应用程序可以简单地请求更新的建议响应。不太可能，但更严重的是，结果可能不同，因为链代码是不确定性的。不确定性是链码和分类账的敌人，如果出现这种情况，则表明所提议的交易存在严重问题，因为不一致的结果显然不能应用于分类账。单个对等机无法知道其事务结果是非确定性的-在检测到非确定性之前，必须将事务响应收集在一起进行比较。（严格来说，即使这还不够，但我们将讨论推迟到事务部分，在这里详细讨论了非确定性。）

At the end of phase 1, the application is free to discard inconsistent transaction responses if it wishes to do so, effectively terminating the transaction workflow early. We' ll see later that if an application tries to use an inconsistent set of transaction responses to update the ledger, it will be rejected.

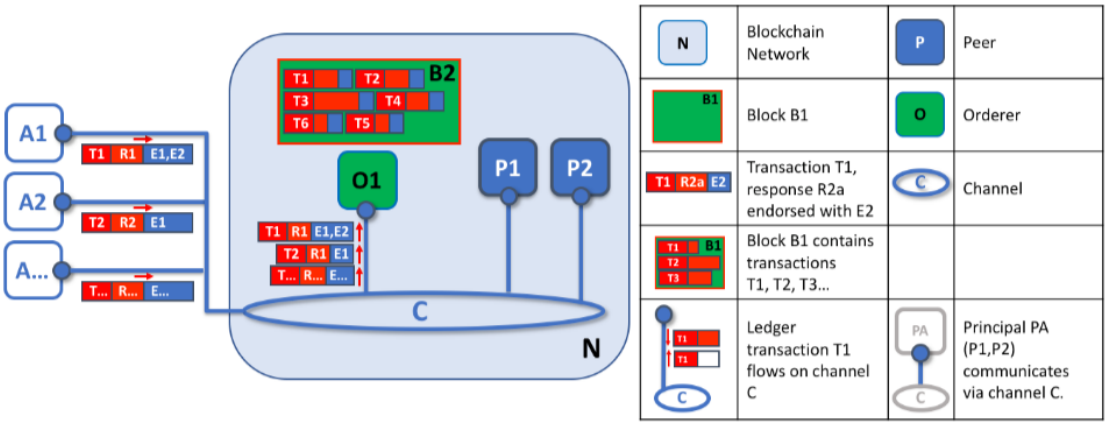
在第 1 阶段结束时，如果希望这样做，应用程序可以自由地丢弃不一致的事务响应，从而有效地提前终止事务工作流。稍后我们将看到，如果应用程序试图使用一组不一致的事务响应来更新分类帐，它将被拒绝。

**Phase 2: Packaging**

**第 2 阶段：包装**

The second phase of the transaction workflow is the packaging phase. The orderer is pivotal to this process — it receives transactions containing endorsed transaction proposal responses from many applications. It orders each transaction relative to other transactions, and packages batches of transactions into blocks ready for distribution back to all peers connected to the orderer, including the original endorsing peers.

交易工作流的第二个阶段是打包阶段。订购方是这个过程的关键-它接收包含来自许多应用程序的已批准交易建议响应的事务。它相对于其他交易对每个交易进行排序，并将成批的交易打包成块，以便分发回与订购方连接的所有对等方，包括原始的认可对等方。



The first role of an orderer node is to package proposed ledger updates. In this example, application A1 sends a transaction T1 endorsed by E1 and E2 to the orderer O1. In parallel, Application A2 sends transaction T2 endorsed by E1 to the orderer O1. O1 packages transaction T1 from application A1 and transaction T2 from application A2 together with other transactions from other applications in the network into block B2. We can see that in B2, the transaction order is T1,T2,T3,T4,T6,T5 - which may not be the order in which these transactions arrived at the orderer node! (This example shows a very simplified orderer configuration.)

orderer 节点的第一个角色是打包建议的分类帐更新。在本例中，应用程序 A1 将由 e1 和 e2 背书的事务 T1 发送给订购方 O1。同时，应用程序 A2 将由 e1 签署的事务 t2 发送给订购方 o1。O1 将来自应用程序 A1 的事务 T1 和来自应用程序 A2 的事务 T2 以及来自网络中其他应用程序的其他事务打包到块 B2 中。我们可以看到，在 b2 中，事务顺序是 t1、t2、t3、t4、t6、t5——这可能不是这些事务到达订购方节点的顺序！（此示例显示了一个非常简化的医嘱者配置。）

An orderer receives proposed ledger updates concurrently from many different applications in the network on a particular channel. Its job is to arrange these proposed updates into a well-defined sequence, and package them into blocks for subsequent distribution. These blocks will become the blocks of the blockchain! Once an orderer has generated a block of the desired size, or after a maximum elapsed time, it will be sent to all peers connected to it on a particular channel. We'll see how this block is processed in phase 3.

订购方在特定通道上同时接收来自网络中许多不同应用程序的建议分类帐更新。它的工作是将这些建议的更新安排到一个定义良好的序列中，并将它们打包成块以供后续分发。这些区块将成为区块链的区块！一旦订购方生成了所需大小的块，或者经过了最长时间后，它将发送到特定通道上连接到它的所有对等端。我们将在第 3 阶段看到如何处理这个块。

It's worth noting that the sequencing of transactions in a block is not necessarily the same as the order of arrival of transactions at the orderer! Transactions can be packaged in any order into a block, and it's this sequence that becomes the order of execution. What's important is that there is a strict order, rather than what that order is.

值得注意的是，块中交易的顺序不一定与交易到达订购方的顺序相同！交易可以按任意顺序打包成一个块，正是这个顺序成为了执行顺序。重要的是有一个严格的顺序，而不是那个顺序。

This strict ordering of transactions within blocks makes Hyperledger Fabric a little different from other blockchains where the same transaction can be packaged into multiple different blocks. In Hyperledger Fabric, this cannot happen — the blocks generated by a collection of orderers are said to be final because once a transaction has been written to a block, its position in the ledger is immutably assured. Hyperledger Fabric's finality means that a disastrous occurrence known as a ledger fork cannot occur. Once transactions are captured in a block, history cannot be rewritten for that

transaction at a future point in time.

这种对块内交易的严格排序使得 Hyperledger Fabric 与其他块链稍有不同，在这些块链中，同一交易可以打包成多个不同的块。在 Hyperledger 结构中，这种情况是不可能发生的——订购者集合生成的块被认为是最终的，因为一旦一个交易被写入一个块，它在分类帐中的位置就不会改变。超级账本结构的最终性意味着一个灾难性的事件，即所谓的账本叉不能发生。一旦在块中捕获了交易，就不能在将来的某个时间点为该事务重写历史记录。

We can see also see that, whereas peers host the ledger and chaincodes, orderers most definitely do not. Every transaction that arrives at an orderer is mechanically packaged in a block — the orderer makes no judgement as to the value of a transaction, it simply packages it. That's an important property of Hyperledger Fabric — all transactions are marshalled into a strict order — transactions are never dropped or de-prioritized.

我们还可以看到，虽然对等方托管分类账和链码，但订购方绝对不会。到达订购方的每个事务都被机械地打包在一个块中——订购方对交易的价值没有任何判断，它只是打包。这是 Hyperledger Fabric 的一个重要属性-所有交易都被编组成一个严格的顺序-交易不会被删除或取消优先级。

At the end of phase 2, we see that orderers have been responsible for the simple but vital processes of collecting proposed transaction updates, ordering them, packaging them into blocks, ready for distribution.

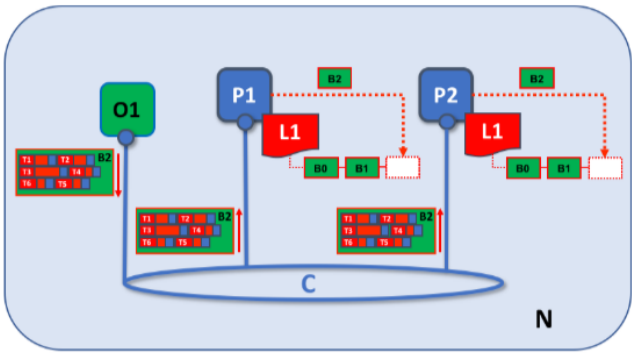
在第 2 阶段结束时，我们看到订购者负责收集提议的交易更新、订购、将它们打包成块、准备分发等简单但重要的过程。

**Phase 3: Validation**

**第 3 阶段：验证**

The final phase of the transaction workflow involves the distribution and subsequent validation of blocks from the orderer to the peers, where they can be applied to the ledger. Specifically, at each peer, every transaction within a block is validated to ensure that it has been consistently endorsed by all relevant organizations before it is applied to the ledger. Failed transactions are retained for audit, but are not applied to the ledger.

交易处理工作流的最后一个阶段涉及从订购方到对等方的块的分发和后续验证，这些块可以应用于分类帐。具体地说，在每个对等方，一个块中的每个事务都经过验证，以确保在将其应用到分类帐之前，所有相关组织都一致认可它。失败的交易记录保留用于审核，但不应用于分类帐。




| | | | |
|---|---|---|---|
| N | Blockchain Network | P | Peer |
| C | Channel | O | Orderer |
| L | Ledger | B | Block B |
| L1 B0 B1 | Ledger L1 has blockchain with blocks B0, B1 | T1 T2 T3 B1 | Block B1 contains transactions T1, T2, T3... |
| B1 C | Block B1 flows on channel C | PA C | Principal PA (P1, P2) communicates via channel C. |

The second role of an orderer node is to distribute blocks to peers. In this example, orderer O1 distributes block B2 to peer P1 and peer P2. Peer P1 processes block B2, resulting in a new block being added to ledger L1 on P1. In parallel, peer P2 processes block B2, resulting in a new block being added to ledger L1 on P2. Once this process is complete, the ledger L1 has been consistently updated on peers P1 and P2, and each may inform connected applications that the transaction has been processed.

排序器节点的第二个角色是将块分发给对等方。在本例中，订购者 O1 将块 B2 分配给对等机 P1 和对等机 P2。对等机 p1 处理块 b2，从而将新块添加到 p1 上的分类帐 l1。并行地，对等机 p2 处理块 b2，导致一个新块被添加到 p2 上的分类帐 l1。一旦这个过程完成，分类账 L1 已经在对等方 P1 和 P2 上得到一致的更新，并且每个人都可以通知相关的应用程序交易已经被处理。

Phase 3 begins with the orderer distributing blocks to all peers connected to it. Peers are connected to orderers on channels such that when a new block is generated, all of the peers connected to the orderer will be sent a copy of the new block. Each peer will process this block independently, but in exactly the same way as every other peer on the channel. In this way, we'll see that the ledger can be kept consistent. It's also worth noting that not every peer needs to be connected to an orderer — peers can cascade blocks to other peers using the gossip protocol, who also can process them independently. But let's leave that discussion to another time!

阶段 3 从订购方将块分配给与其连接的所有对等方开始。对等端连接到通道上的医嘱者，这样当生成新块时，所有连接到医嘱者的对等端都将发送新块的副本。每个对等端将独立地处理此块，但方式与通道上的其他对等端完全相同。这样，我们就可以看到分类帐可以保持一致。同样值得注意的是，并不是每个对等点都需要连接到一个订购者，对等点可以使用八卦协议将块级联到其他对等点，后者也可以独立处理它们。但让我们把讨论留给另一个时间吧！

Upon receipt of a block, a peer will process each transaction in the sequence in which it appears in the block. For every transaction, each peer will verify that the transaction has been endorsed by the required organizations according to the endorsement policy of the chaincode which generated the transaction. For example, some transactions may only need to be endorsed by a single organization, whereas others may require multiple endorsements before they are considered valid. This process of validation verifies that all relevant organizations have generated the same outcome or result. Also note that this validation is different than the endorsement check in phase 1, where it is the application that receives the response from endorsing peers and makes the decision to send the proposal transactions. In case the application violates the endorsement policy by sending wrong transactions, the peer is still able to reject the transaction in the validation process of phase 3.

在收到一个块后，对等方将按照它在块中出现的顺序处理每个事务。对于每个事务，每个对等方都将根据生成该事务的链码的批核策略验证该事务是否已由所需组织进行批核。例如，某些交易可能只需要由一个组织背书，而其他交易可能需要多个背书才能

被视为有效。此验证过程验证所有相关组织是否生成了相同的结果或结果。还请注意，此验证与阶段 1 中的认可检查不同,在阶段 1 中,应用程序接收来自认可对等方的响应，并决定发送建议事务。如果应用程序违反了认可策略，发送了错误的事务，则对等方仍然能够在第 3 阶段的验证过程中拒绝该事务。

If a transaction has been endorsed correctly, the peer will attempt to apply it to the ledger. To do this, a peer must perform a ledger consistency check to verify that the current state of the ledger is compatible with the state of the ledger when the proposed update was generated. This may not always be possible, even when the transaction has been fully endorsed. For example, another transaction may have updated the same asset in the ledger such that the transaction update is no longer valid and therefore can no longer be applied. In this way each peer's copy of the ledger is kept consistent across the network because they each follow the same rules for validation.

如果交易已正确背书，对等方将尝试将其应用于分类帐。为此，对等方必须执行分类帐一致性检查，以验证生成建议更新时分类帐的当前状态是否与分类帐的状态兼容。这可能并不总是可能的，即使交易已经完全背书。例如，另一个交易可能更新了分类帐中的同一资产，因此交易更新不再有效，因此无法再应用。这样，每个对等方的分类账副本在整个网络中保持一致，因为它们都遵循相同的验证规则。

After a peer has successfully validated each individual transaction, it updates the ledger. Failed transactions are not applied to the ledger, but they are retained for audit purposes, as are successful transactions. This means that peer blocks are almost exactly the same as the blocks received from the orderer, except for a valid or invalid indicator on each transaction in the block.

对等方成功验证每个交易后，它将更新分类帐。失败的交易不应用于分类帐，但它们保留用于审计目的，成功的交易也是如此。这意味着，除了块中每个事务的有效或无效指示器外，对等块几乎与从订购方接收的块完全相同。

We also note that phase 3 does not require the running of chaincodes — this is done only during phase 1, and that's important. It means that chaincodes only have to be available on endorsing nodes, rather than throughout the blockchain network. This is often helpful as it keeps the logic of the chaincode confidential to endorsing organizations. This is in contrast to the output of the chaincodes (the transaction proposal responses) which are shared with every peer in the channel, whether or not they endorsed the transaction. This specialization of endorsing peers is designed to help scalability.

我们还注意到，阶段 3 不需要运行链码——这只在阶段 1 中完成，这很重要。这意味着链码只能在认可节点上使用，而不能在整个区块链网络上使用。这通常是有帮助的，因为它对认可组织的链式代码的逻辑保密。这与链接代码（事务建议响应）的输出相反，这些链接代码与通道中的每个对等端共享，不管它们是否认可该事务。这种认可对等点的专门化设计是为了帮助可扩展性。

Finally, every time a block is committed to a peer's ledger, that peer generates an appropriate event. Block events include the full block content,

while block transaction events include summary information only, such as whether each transaction in the block has been validated or invalidated. Chaincode events that the chaincode execution has produced can also be published at this time. Applications can register for these event types so that they can be notified when they occur. These notifications conclude the third and final phase of the transaction workflow.

最后，每次将块提交到对等方的分类帐时，该对等方都会生成适当的事件。块事件包括完整的块内容，而块交易事件仅包括摘要信息，例如块中的每个交易是否已验证或失效。此时也可以发布执行 chaincode 所生成的 chaincode 事件。应用程序可以注册这些事件类型，以便在它们发生时得到通知。这些通知结束交易工作流的第三个和最后一个阶段。

In summary, phase 3 sees the blocks which are generated by the orderer consistently applied to the ledger. The strict ordering of transactions into blocks allows each peer to validate that transaction updates are consistently applied across the blockchain network.

总之，第 3 阶段将看到订购方生成的块，这些块始终应用于分类账。将交易严格排序成块允许每个对等方验证在区块链网络中一致应用交易更新。

**Orderers and Consensus**

**订购方和协商一致**

This entire transaction workflow process is called consensus because all peers have reached agreement on the order and content of transactions, in a process that is mediated by orderers. Consensus is a multi-step process and applications are only notified of ledger updates when the process is complete — which may happen at slightly different times on different peers.

这整个事务工作流过程被称为共识，因为所有对等方都已经在一个由订购方调解的过程中就事务的顺序和内容达成了一致。共识是一个多步骤的过程，只有当流程完成时，才会通知应用程序分类帐更新-这可能在不同的同行的不同时间发生。

We will discuss orderers in a lot more detail in a future orderer topic, but for now, think of orderers as nodes which collect and distribute proposed ledger updates from applications for peers to validate and include on the ledger.

我们将在未来的订购者主题中更详细地讨论订购者，但现在，将订购者视为从应用程序收集和分发建议的分类帐更新的节点，以便同行验证并包括在分类帐中。

That’s it! We’ve now finished our tour of peers and the other components that they relate to in Hyperledger Fabric. We’ve seen that peers are in many ways the most fundamental element — they form the network, host chaincodes and the ledger, handle transaction proposals and responses, and keep the ledger up-to-date by consistently applying transaction updates to it.

就是这样！我们现在已经完成了同行和他们在 Hyperledger 结构中相关的其他组件的访问。我们已经看到，同行在许多方面都是最基本的要素——它们形成网络、主机链代码和分类账，处理交易建议和响应，并通过不断地向其应用交易更新来保持分类账的最新状态。

# 八、Private data

# 八、私有数据

## 1、What is private data?

## 1、什么是私有数据？

In cases where a group of organizations on a channel need to keep data private from other organizations on that channel, they have the option to create a new channel comprising just the organizations who need access to the data. However, creating separate channels in each of these cases creates additional administrative overhead (maintaining chaincode versions, policies, MSPs, etc), and doesn't allow for use cases in which you want all channel participants to see a transaction while keeping a portion of the data private.

如果一个渠道上的一组组织需要对该渠道上的其他组织的数据保密，他们可以选择创建一个新的渠道，该渠道只包括需要访问数据的组织。但是，在每种情况下创建单独的通道都会产生额外的管理开销（维护链码版本、策略、MSP 等），并且不允许使用希望所有通道参与者在保持部分数据私有的情况下看到交易的用例。

That's why, starting in v1.2, Fabric offers the ability to create private data collections, which allow a defined subset of organizations on a channel the ability to endorse, commit, or query private data without having to create a separate channel.

这就是为什么，从 v1.2 开始，Fabric 提供了创建私有数据集合的能力，这允许在一个通道上定义的组织子集在不需要创建单独通道的情况下认可、提交或查询私有数据。

## 2、What is a private data collection?

## 2、什么是私有数据收集？

A collection is the combination of two elements:

集合是两个元素的组合：

1、The actual private data, sent peer-to-peer via gossip protocol to only the organization(s) authorized to see it. This data is stored in a private database on the peer (sometimes called a "side" database, or "SideDB"). The ordering service is not involved here and does not see the private data. Note that setting up gossip requires setting up anchor peers in order to bootstrap cross-organization communication.

1、实际的私有数据，通过 gossip 协议发送到只有被授权查看它的组织。这些数据存储在对等端的私有数据库中（有时称为“side”数据库或“sidedb”）。这里不涉及订购服务，也看不到私有数据。请注意，设置 gossip 需要设置锚点对等以便引导跨组织通信。

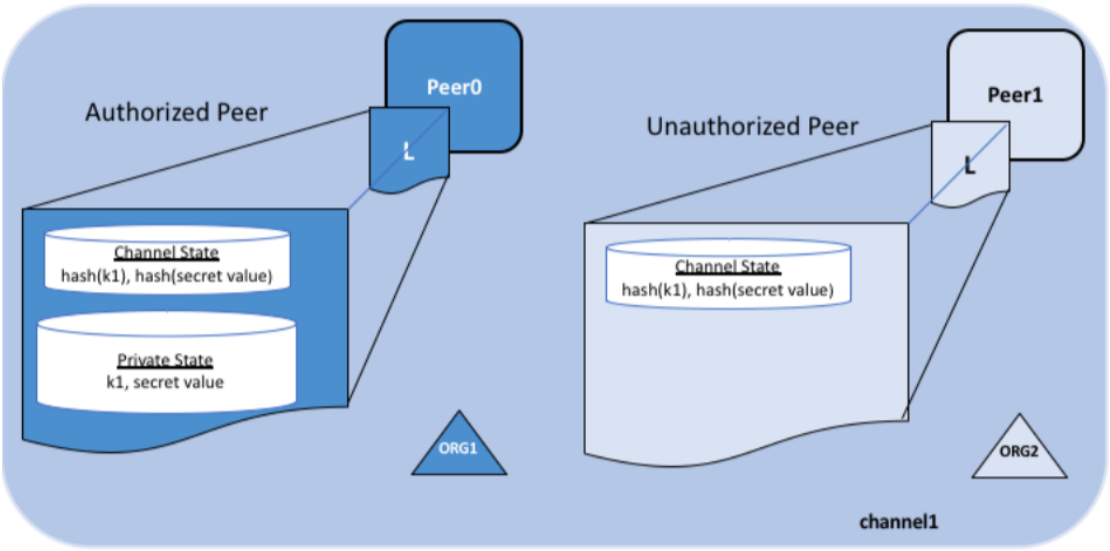
2、A hash of that data, which is endorsed, ordered, and written to the ledgers of every peer on the channel. The hash serves as evidence of the transaction and is used for state validation and can be used for audit purposes.

2、该数据的散列，它被背书、排序并写入通道上每个对等方的分类账中。哈希用作

交易的证据，用于状态验证，可以用于审计目的。

The following diagram illustrates the ledger contents of a peer authorized to have private data and one which is not.

下图说明了被授权拥有私有数据的对等方和未被授权拥有私有数据的对等方的分类帐内容。




Collection members may decide to share the private data with other parties if they get into a dispute or if they want to transfer the asset to a third party. The third party can then compute the hash of the private data and see if it matches the state on the channel ledger, proving that the state existed between the collection members at a certain point in time.

如果收集成员遇到争议或想要将资产转让给第三方，他们可以决定与其他方共享私人数据。然后，第三方可以计算私有数据的散列值，并查看它是否与通道分类账上的状态匹配，从而证明在某个时间点收集成员之间存在状态。

**When to use a collection within a channel vs. a separate channel**

**何时在通道内使用集合与单独的通道**

- Use channels when entire transactions (and ledgers) must be kept confidential within a set of organizations that are members of the channel.
- Use collections when transactions (and ledgers) must be shared among a set of organizations, but when only a subset of those organizations should have access to some (or all) of the data within a transaction. Additionally, since private data is disseminated peer-to-peer rather than via blocks, use private data collections when transaction data must be kept confidential from ordering service nodes.
- 当整个交易（和分类账）必须在属于渠道成员的一组组织中保密时，使用渠道。
- 当交易（和分类帐）必须在一组组织之间共享，但只有这些组织的一个子集可以访问交易中的某些（或全部）数据时，使用集合。此外，由于私有数据是通过点对点

而不是通过块传播的，因此在必须对事务数据保密的情况下，使用私有数据收集来订购服务节点。

## 3、A use case to explain collections

## 3、用于解释集合的用例

Consider a group of five organizations on a channel who trade produce:
考虑一个渠道上有五个组织从事农产品贸易：

- **A Farmer** selling his goods abroad
- **A Distributor** moving goods abroad
- **A Shipper** moving goods between parties
- **A Wholesaler** purchasing goods from distributors
- **A Retailer** purchasing goods from shippers and wholesalers
- 在国外卖东西的农民
- 将货物运到国外的分销商
- 在双方之间移动货物的托运人
- 从经销商处采购货物的批发商
- 向托运人和批发商购买货物的零售商

The Distributor might want to make private transactions with the Farmer and Shipper to keep the terms of the trades confidential from the Wholesaler and the Retailer (so as not to expose the markup they’re charging).
经销商可能希望与农场主和发货人进行私人交易，以对批发商和零售商的交易条款保密（以免暴露他们收取的加价）。

The Distributor may also want to have a separate private data relationship with the Wholesaler because it charges them a lower price than it does the Retailer.
分销商也可能希望与批发商建立单独的私人数据关系，因为它向批发商收取的价格比零售商低。

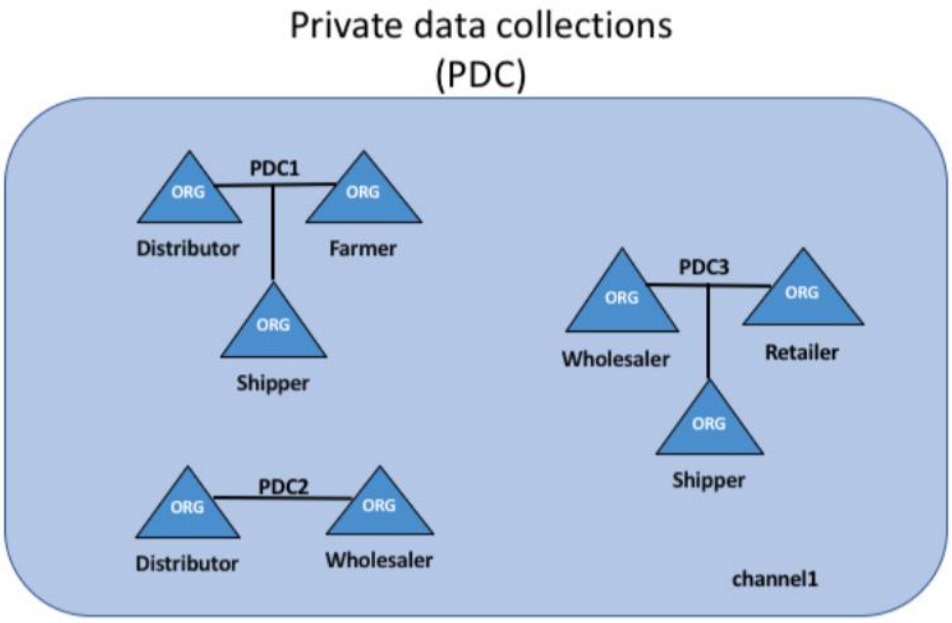
The Wholesaler may also want to have a private data relationship with the Retailer and the Shipper.
批发商也可能希望与零售商和发货人建立私人数据关系。

Rather than defining many small channels for each of these relationships, multiple private data collections (PDC) can be defined to share private data between:
可以定义多个私有数据收集（pdc）来共享以下之间的私有数据，而不是为每个关系定义多个小通道：

1、PDC1: Distributor, Farmer and Shipper
2、PDC2: Distributor and Wholesaler
3、PDC3: Wholesaler, Retailer and Shipper
1、PDC1：经销商、农场主和发货人

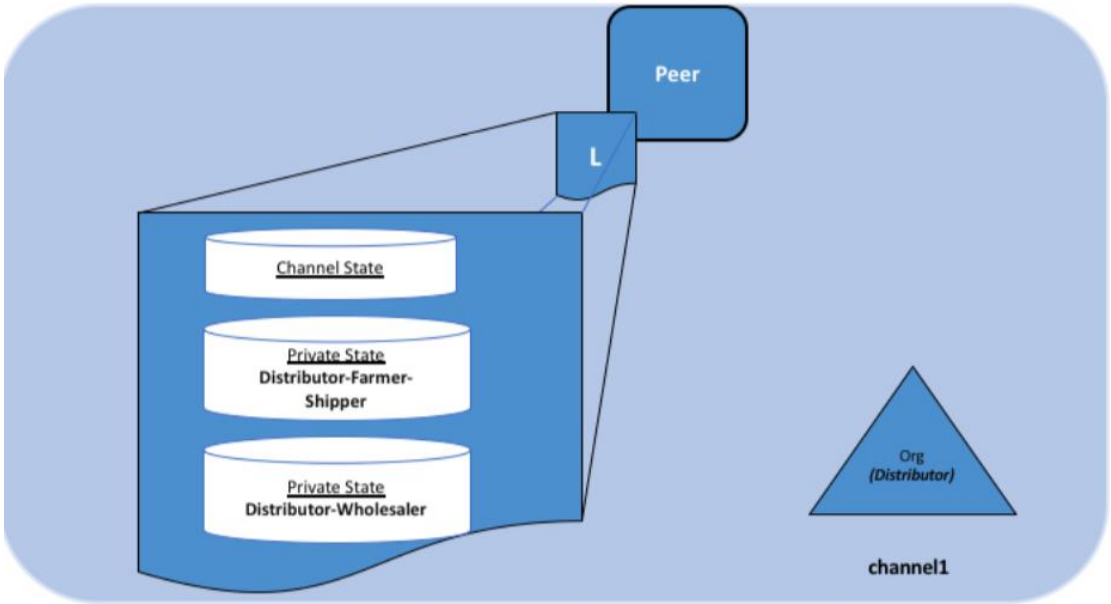
2、PDC2：经销商和批发商

3、PDC3：批发商、零售商和发货人




Using this example, peers owned by the Distributor will have multiple private databases inside their ledger which includes the private data from the Distributor, Farmer and Shipper relationship and the Distributor and Wholesaler relationship. Because these databases are kept separate from the database that holds the channel ledger, private data is sometimes referred to as “SideDB”.

使用此示例，分销商拥有的对等方在其分类账中会有多个私有数据库，其中包括来自分销商、农民和发货人关系以及分销商和批发商关系的私有数据。因为这些数据库与持有渠道分类账的数据库是分开的，所以私有数据有时被称为“sidedb”。




## 5、Transaction flow with private data

## 5、具有私有数据的交易流

When private data collections are referenced in chaincode, the transaction flow is slightly different in order to protect the confidentiality of the private data as transactions are proposed, endorsed, and committed to the ledger.

当私有数据收集在 chaincode 中被引用时，为了保护私有数据的机密性，交易流程略有不同，因为交易被提议、批准并提交到分类帐。

For details on transaction flows that don’t use private data refer to our documentation on transaction flow.

有关不使用私有数据的交易流的详细信息，请参阅我们有关交易流的文档。

1、The client application submits a proposal request to invoke a chaincode function (reading or writing private data) to endorsing peers which are part of authorized organizations of the collection. The private data, or data used to generate private data in chaincode, is sent in a transient field of the proposal.

1、客户机应用程序提交了一个建议请求，以调用作为集合授权组织一部分的对等方的链码函数（读取或写入私有数据）。私有数据，或用于在链码中生成私有数据的数据，在提案的临时字段中发送。

2、The endorsing peers simulate the transaction and store the private data in a transient data store (a temporary storage local to the peer). They distribute the private data, based on the collection policy, to authorized peers via gossip.

2、认可对等机模拟交易，并将私有数据存储在临时数据存储区（对等机本地的临时存储区）中。他们根据收集政策，通过流言蜚语将私有数据分发给授权的对等方。

3、The endorsing peer sends the proposal response back to the client with public data, including a hash of the private data key and value. No private data is sent back to the client. For more information on how endorsement works with private data, click here.

3、认可对等端使用公共数据（包括私有数据密钥和值的散列）将建议响应发送回客户机。没有私人数据发送回客户端。有关认可如何处理私人数据的详细信息，请单击此处。

4、The client application submits the transaction to the ordering service (with hashes of the private data) which gets distributed into blocks as normal. The block with the hashed values is distributed to all the peers. In this way, all peers on the channel can validate transactions with the hashes of the private data in a consistent way, without knowing the actual private data.

4、客户机应用程序将交易提交给排序服务（使用私有数据的散列），这些散列按正常方式分布到块中。具有散列值的块被分发到所有对等方。这样，通道上的所有对等方都可以一致地使用私有数据的散列验证事务，而不必知道实际的私有数据。

5、At block-committal time, authorized peers use the collection policy to determine if they are authorized to have access to the private data. If they do, they will first check their local transient data store to determine if

they have already received the private data at chaincode endorsement time. If not, they will attempt to pull the private data from another peer. Then they will validate the private data against the hashes in the public block and commit the transaction and the block. Upon validation/commit, the private data is moved to their copy of the private state database and private writeset storage. The private data is then deleted from the transient data store.

5、在块提交时，授权对等方使用收集策略来确定他们是否被授权访问私有数据。如果他们这样做了，他们将首先检查他们的本地瞬态数据存储，以确定他们是否已经在链码认可时收到了私有数据。如果没有，他们将尝试从另一个对等机中提取私有数据。然后，他们将根据公共块中的散列验证私有数据，并提交事务和块。验证/提交后，私有数据将移动到私有状态数据库和私有写集存储的副本中。然后从临时数据存储中删除私有数据。

## 6、How a private data collection is defined

## 6、如何定义私有数据收集

For more details on collection definitions, and other low level information about private data and collections, refer to the private data reference topic.

有关收集定义以及其他有关私有数据和收集的低级信息的详细信息，请参阅私有数据引用主题。

## 7、Purging data

## 7、清除数据

For very sensitive data, even the parties sharing the private data might want — or might be required by government regulations — to “purge” the data stored on their peers after a set amount of time, leaving behind only a hash of the data to serve as immutable evidence of the transaction.

对于非常敏感的数据，即使是共享私人数据的各方，也可能希望（或可能是政府法规要求的）在一段时间后“清除”存储在其对等方上的数据，只留下一堆数据作为交易的不可变证据。

In some of these cases, the private data only needs to exist on the peer’s private database until it can be replicated into a database external to the blockchain network. The data might also only need to exist on the peers until a chaincode business process is done with it (trade settled, contract fulfilled, etc). To support the later use case, it is possible to purge private data if it has not been modified once a set number of subsequent blocks have been added to the private database.

在某些情况下，私有数据只需要存在于对等方的私有数据库中，直到它可以复制到区块链网络外部的数据库中。数据也可能只需要存在于对等端上，直到完成了链码业务流程（贸易结算、合同履行等）。为了支持以后的用例，如果在私有数据库中添加了一定数量的后续块后，没有对私有数据进行修改，则可以清除私有数据。

# 九、Ledger

# 九、分类账

## 1、What is a Ledger?

## 1、什么是分类帐？

A ledger contains the current state of a business as a journal of transactions. The earliest European and Chinese ledgers date from almost 1000 years ago, and the Sumerians had stone ledgers 4000 years ago – but let's start with a more up-to-date example!

分类帐包含作为交易日记帐的业务的当前状态。最早的欧洲和中国分类账可追溯到大约 1000 年前，而苏美尔人有 4000 年前的石头分类账——但让我们从一个更为新的例子开始吧！

You're probably used to looking at your bank account every month. What's most important to you is the available balance – it's what you're able to spend at the current moment in time. If you want to see how your balance was derived, then you can look through the transaction credits and debits that determined it. This is a real life example of a ledger – a state (your bank balance), and a set of ordered transactions (credits and debits) that determine it. Hyperledger Fabric is motivated by these same two concerns – to present the current value of a set of ledger states, and to capture the history of the transactions that determined these states.

你可能习惯于每个月查看你的银行账户。对你来说最重要的是可用的余额——它是你在当前时刻能够花费的时间。如果您想了解余额是如何得出的，那么您可以查看确定余额的交易贷记和借记。这是一个真实的分类账例子——一个状态（您的银行余额），以及一组确定它的有序交易（贷记和借记）。Hyperledger 结构受到这两个相同的关注点的激励——呈现一组分类帐状态的当前值，并捕获确定这些状态的交易的历史记录。

Let's take a closer look at the Hyperledger Fabric ledger structure!

让我们更仔细地看一下 Hyperledger 结构分类账！

## 2、A Blockchain Ledger

## 2、区块链分类账

A blockchain ledger consists of two distinct, though related, parts – a world state and a blockchain.

区块链分类账由两个不同但相关的部分组成——一个世界状态和一个区块链。

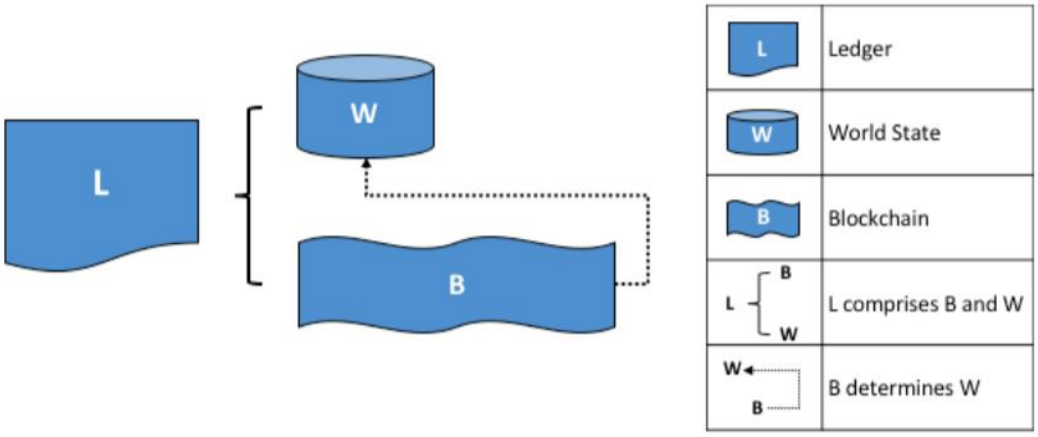
Firstly, there's a world state – a database that holds the current values of a set of ledger states. The world state makes it easy for a program to get the current value of these states, rather than having to calculate them by traversing the entire transaction log. Ledger states are, by default, expressed as key-value pairs, though we'll see later that Hyperledger Fabric provides flexibility in this regard. The world state can change frequently, as states can be created, updated and deleted.

首先，有一个世界状态——一个保存一组分类帐状态当前值的数据库。世界状态使

得程序很容易获得这些状态的当前值，而不是通过遍历整个事务日志来计算它们。默认情况下，分类帐状态表示为键值对，不过稍后我们将看到，Hyperledger 结构在这方面提供了灵活性。世界状态可以经常更改，因为状态可以被创建、更新和删除。

Secondly, there's a blockchain – a transaction log that records all the changes that determine the world state. Transactions are collected inside blocks that are appended to the blockchain – enabling you to understand the history of changes that have resulted in the current world state. The blockchain data structure is very different to the world state because once written, it cannot be modified. It is an immutable sequence of blocks, each of which contains a set of ordered transactions.

其次，有一个区块链——一个记录决定世界状态的所有变化的交易日志。交易被收集在附加到区块链的块中，使您能够了解导致当前世界状态的变化历史。区块链数据结构与世界状态非常不同，因为一旦写入，就无法修改。它是一个不可变的块序列，每个块包含一组有序交易。




The visual vocabulary expressed in facts is as follows: Ledger L comprises blockchain B and World State W. Blockchain B determines World State W. Also expressed as: World state W is derived from blockchain B.

事实表达的视觉词汇如下：分类账 L 由区块链 B 和世界状态 W 组成，区块链 B 决定世界状态 W，也表示为：世界状态 W 源于区块链 B。

It's helpful to think of there being one logical ledger in a Hyperledger Fabric network. In reality, the network maintains multiple copies of a ledger – which are kept consistent with every other copy through a process called consensus. The term Distributed Ledger Technology (DLT) is often associated with this kind of ledger – one that is logically singular, but has many consistent copies distributed throughout a network.

在一个超级账本结构网络中有一个逻辑账本是很有帮助的。实际上，网络维护了一个分类账的多个副本——通过一个称为共识的过程，这些副本与其他所有副本保持一致。术语分布式账本技术（DLT）通常与这种账本相关联，这种账本在逻辑上是单一的，但在整个网络中有许多一致的副本。

Let's now examine the world state and blockchain data structures in more detail.

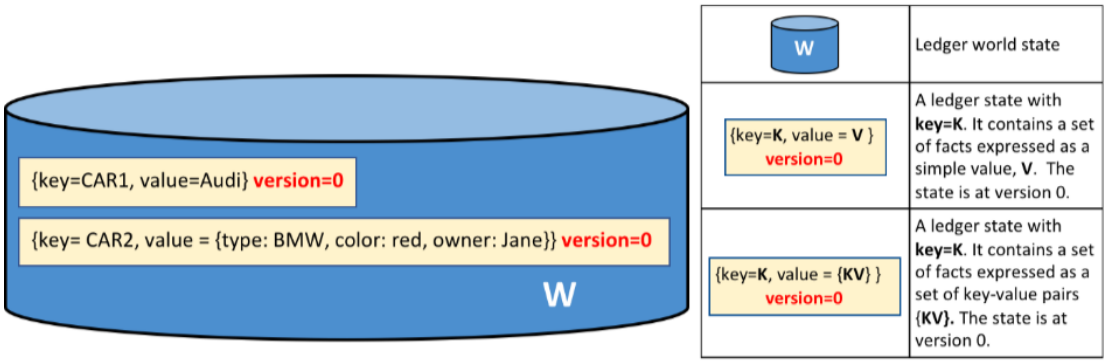
现在让我们更详细地研究世界状态和区块链数据结构。

## 3、World State

## 3、世界状态

The world state represents the current values of all ledger states. It's extremely useful because programs usually need the current value of a ledger state and that's always easily available. You do not need to traverse the entire blockchain to calculate the current value of any ledger state – you just get it directly from the world state.

世界状态表示所有分类帐状态的当前值。它非常有用，因为程序通常需要分类帐状态的当前值，而且总是很容易获得。您不需要遍历整个区块链来计算任何分类账状态的当前值——您只需直接从世界状态获取它。




The visual vocabulary expressed in facts is as follows: There is a ledger state with key=CAR1 and value=Audi. There is a ledger state with key=CAR2 and a more complex value {model:BMW, color=red, owner=Jane}. Both states are at version 0.

用事实表示的可视词汇表如下：有一个分类帐状态，key=car1，value=audi。有一个分类帐状态，key=car2，还有一个更复杂的值模型：宝马，color=red，owner=jane。两个状态都在版本 0 中。

Ledger state is used to record application information to be shared via the blockchain. The example above shows ledger states for two cars, CAR1 and CAR2. You can see that states have a key and a value. Your application programs invoke chaincode which access states via simple APIs – they get, put and delete states using a state key. Notice how a state value can be simple (Audi…) or complex (type:BMW…).

分类帐状态用于记录要通过区块链共享的应用程序信息。上面的示例显示了两辆汽车的分类帐状态，即 car1 和 car2。您可以看到状态有一个键和一个值。您的应用程序调用通过简单的 API 访问状态的链代码——它们使用状态键获取、放置和删除状态。请注意状态值可以是简单的（奥迪…）或复杂的（类型：宝马…）。

Physically, the world state is implemented as a database. This makes a lot of sense because a database provides a rich set of operators for the efficient storage and retrieval of states. We'll see later that Hyperledger Fabric can be configured to use different world state databases to address

the needs of different types of state values and the access patterns required by applications, for example in complex queries.

实际上，世界状态是作为数据库实现的。这是很有意义的，因为数据库提供了一组丰富的运算符，用于有效地存储和检索状态。稍后我们将看到，可以将 Hyperledger 结构配置为使用不同的世界状态数据库来满足不同类型状态值的需求以及应用程序所需的访问模式，例如在复杂查询中。

Transactions capture changes to the world state, and as you'd expect, transactions have a lifecycle. They are created by applications, and finally end up being committed to the ledger blockchain. The whole lifecycle is described in detail here; but the key design point for Hyperledger Fabric is that only transactions that are signed by a set of endorsing organizations will result in an update to the world state. If a transaction is not signed by sufficient endorsers, then it will fail this validity check, and will not result in an update to the world state.

事务捕获世界状态的变化，正如您所期望的，事务具有生命周期。它们是由应用程序创建的，最后被提交到分类账区块链。这里详细描述了整个生命周期；但是 Hyperledger 结构的关键设计点是，只有由一组认可组织签署的事务才会导致对世界状态的更新。如果一个交易没有足够的背书人签署，那么它将无法通过有效性检查，并且不会导致世界状态的更新。

You'll also notice that a state has a version number, and in the diagram above, states CAR1 and CAR2 are at their starting versions, 0. The version number of a state is incremented every time the state changes. It is also checked whenever the state is updated - to make sure it matches the version when the transaction was created. This check ensures that the world state changing from the same expected value to the same expected value as when the transaction was created.

您还将注意到一个状态有一个版本号，在上面的图表中，状态 car1 和 car2 的初始版本是 0。每次状态更改时，状态的版本号都会增加。当状态更新时，也会检查它，以确保它与创建事务时的版本相匹配。此检查确保世界状态从与创建事务时相同的预期值更改为相同的预期值。

Finally, when a ledger is first created, the world state is empty. Because any transaction which represents a valid change to world state is recorded on the blockchain, it means that the world state can be re-generated from the blockchain at any time. This can be very convenient - for example, the world state is automatically generated when a peer is created. Moreover, if a peer fails abnormally, the world state can be regenerated on peer restart, before transactions are accepted.

最后，当第一次创建分类帐时，世界状态为空。因为任何代表世界状态有效变化的交易都记录在区块链上，这意味着世界状态可以随时从区块链中重新生成。这非常方便——例如，当创建对等时，会自动生成世界状态。此外，如果一个对等机出现异常故障，在接受事务之前，可以在对等机重新启动时重新生成世界状态。

## 4、Blockchain

## 4、区块链

Let’s now turn our attention from the ledger world state to the ledger blockchain.

现在让我们将注意力从 Ledger World State 转到 Ledger 区块链。

The blockchain is a transaction log, structured as interlinked blocks, where each block contains a sequence of transactions, each of which represents a query or update to the world state. The exact mechanism by which transactions are ordered is discussed elsewhere – what’s important is that block sequencing, as well as transaction sequencing within blocks, is established when blocks are first created.

区块链是一个交易日志，结构为互联块，其中每个块包含一系列交易，每个交易代表对世界状态的查询或更新。其他地方讨论了事务排序的确切机制——重要的是块排序以及块内的事务排序是在首次创建块时建立的。

Each block’s header includes a hash of the block’s transactions, as well a copy of the hash of the prior block’s header. In this way, all transactions on the ledger are sequenced and cryptographically linked together. This hashing and linking makes the ledger data very secure. Even if one node hosting the ledger was tampered with, it would not be able to convince all the other nodes that it has the ‘correct’ blockchain because the ledger is distributed throughout a network of independent nodes.

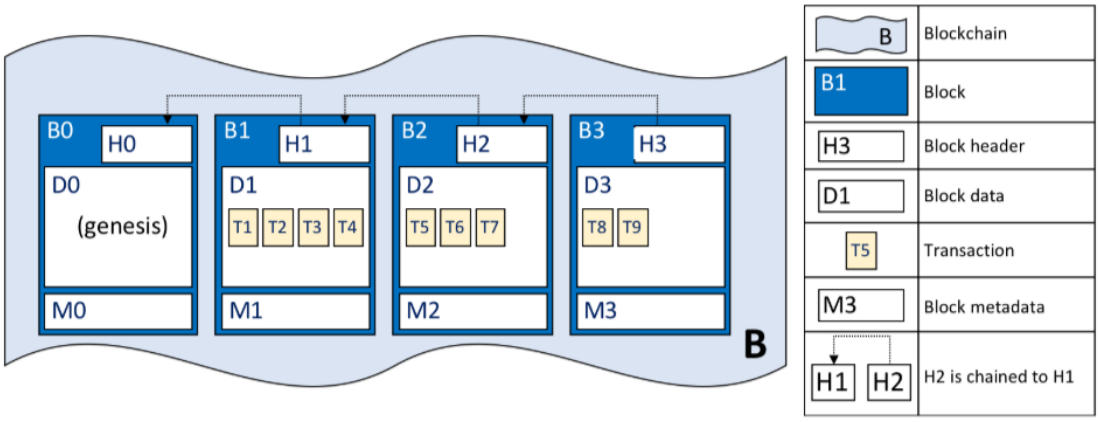
每个块的头包含块事务的散列，以及前一个块头的散列的副本。通过这种方式，分类账上的所有交易都按顺序排列并以密码方式链接在一起。这种散列和链接使得分类帐数据非常安全。即使一个托管分类账的节点被篡改，它也无法说服所有其他节点它拥有“正确”的区块链，因为分类账分布在一个由独立节点组成的网络中。

Physically, the blockchain is always implemented as a file, in contrast to the world state, which uses a database. This is a sensible design choice as the blockchain data structure is heavily biased towards a very small set of simple operations. Appending to the end of the blockchain is the primary operation, and query is currently a relatively infrequent operation.

在物理上，区块链总是作为一个文件来实现，与使用数据库的世界状态不同。这是一个明智的设计选择，因为区块链数据结构严重偏向于一组非常小的简单操作。附加到区块链末尾是主要操作，而查询目前是一个相对较少的操作。

Let’s have a look at the structure of a blockchain in a little more detail.

让我们更详细地了解一下区块链的结构。




The visual vocabulary expressed in facts is as follows: Blockchain B contains blocks B0, B1, B2, B3. B0 is the first block in the blockchain, the genesis block

事实表达的视觉词汇如下：区块链 B 包含区块 b0、b1、b2、b3。b0 是区块链中的第一个区块，Genesis 区块

In the above diagram, we can see that block B2 has a block data D2 which contains all its transactions: T5, T6, T7.

在上图中，我们可以看到块 B2 有一个包含所有事务的块数据 D2：t5、t6、t7。

Most importantly, B2 has a block header H2, which contains a cryptographic hash of all the transactions in D2 as well as with the equivalent hash from the previous block B1. In this way, blocks are inextricably and immutably linked to each other, which the term blockchain so neatly captures!

最重要的是，b2 有一个块头 h2，其中包含 d2 中所有事务的加密哈希，以及前一个块 b1 中的等效哈希。通过这种方式，区块是不可分割的和不可改变的相互连接，区块链这个术语如此巧妙地捕捉到了这一点！

Finally, as you can see in the diagram, the first block in the blockchain is called the genesis block. It’s the starting point for the ledger, though it does not contain any user transactions. Instead, it contains a configuration transaction containing the initial state of the network channel (not shown). We discuss the genesis block in more detail when we discuss the blockchain network and channels in the documentation.

最后，如图中所示，区块链中的第一个区块称为 Genesis 区块。它是分类帐的起点，尽管它不包含任何用户事务。相反，它包含一个包含网络通道初始状态的配置事务（未显示）。当我们在文档中讨论区块链网络和渠道时，我们更详细地讨论 Genesis 块。

## 5、Blocks

## 5、区块

Let’s have a closer look at the structure of a block. It consists of three sections

让我们仔细看看一个块的结构。它由三部分组成

**Block Header**

**区块头**

This section comprises three fields, written when a block is created.

本节包含三个字段，在创建块时写入。

Block number: An integer starting at 0 (the genesis block), and increased by 1 for every new block appended to the blockchain.

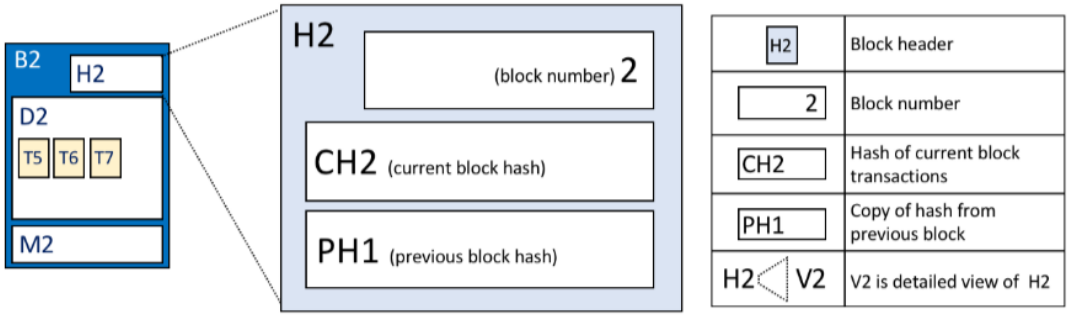
块号：一个从 0 开始的整数（Genesis 块），每个附加到区块链的新块增加 1 个。

Current Block Hash: The hash of all the transactions contained in the current block.

当前块哈希：当前块中包含的所有事务的哈希。

Previous Block Hash: A copy of the hash from the previous block in the blockchain.

上一个区块哈希：区块链上一个区块的哈希副本




The visual vocabulary expressed in facts is as follows: Block header H2 of block B2 consists of block number 2, the hash CH2 of the current block data D2, and a copy of a hash PH1 from the previous block, block number 1.

用真实表示的可视化词汇表如下：块 B2 的块头 h2 由块号 2、当前块数据 d2 的 hash ch2 和上一块块的 hash ph1 的副本 1 组成。

**Block Data**

**块数据**

This section contains a list of transactions arranged in order. It is written when the block is created. These transactions have a rich but straightforward structure, which we describe later in this topic.

此部分包含按顺序排列的事务列表。它是在创建块时写入的。这些事务具有丰富但简单的结构，我们稍后将在本主题中介绍。

**Block Metadata**

**块元数据**

This section contains the time when the block was written, as well as the certificate, public key and signature of the block writer. Subsequently, the block committer also adds a valid/invalid indicator for every transaction, though this information is not included in the hash, as that is created when the block is created.

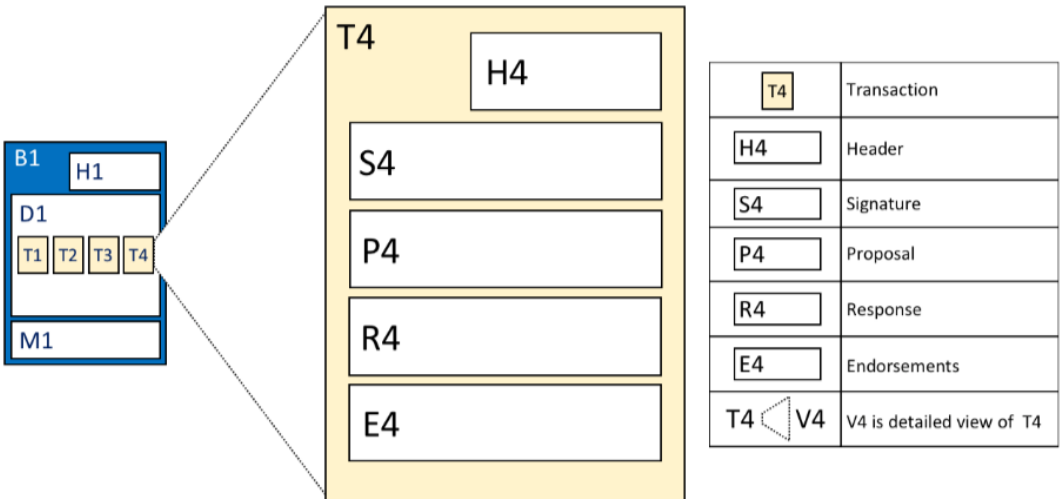
此部分包含块的写入时间，以及块写入程序的证书、公钥和签名。随后，块提交者还为每个事务添加了一个有效/无效的指示器，尽管散列中不包括此信息，因为散列是在创建块时创建的。

## 6、Transactions

## 6、交易

As we’ve seen, a transaction captures changes to the world state. Let’s have a look at the detailed blockdata structure which contains the transactions in a block.

正如我们所看到的，事务捕捉到世界状态的变化。让我们看看详细的 blockdata 结构，它包含一个块中的事务。




The visual vocabulary expressed in facts is as follows: Transaction T4 in blockdata D1 of block B1 consists of transaction header, H4, a transaction signature, S4, a transaction proposal P4, a transaction response, R4, and a list of endorsements, E4.

事实表达的可视化词汇表如下：B1 块的 blockdata d1 中的事务 t4 由事务头、h4、事务签名、s4、事务建议 p4、事务响应、r4 和背书列表 e4 组成。

In the above example, we can see the following fields:

在上面的示例中，我们可以看到以下字段：

**Header**

**头**

This section, illustrated by H4, captures some essential metadata about the transaction - for example, the name of the relevant chaincode, and its version.

本节由 h4 说明，捕获有关事务的一些基本元数据，例如相关链码的名称及其版本。

**Signature**

**签名**

This section, illustrated by S4, contains a cryptographic signature, created by the client application. This field is used to check that the transaction details have not been tampered with, as it requires the application’s private key to generate it.

本节由 S4 说明，包含由客户端应用程序创建的加密签名。此字段用于检查事务详细信息是否被篡改，因为它需要应用程序的私钥来生成它。

**Proposal**

**建议**

This field, illustrated by P4, encodes the input parameters supplied by an application to the chaincode which creates the proposed ledger update. When the chaincode runs, this proposal provides a set of input parameters, which, in combination with the current world state, determines the new world state.

此字段由 P4 说明，将应用程序提供的输入参数编码为创建建议的分类帐更新的链代码。当链码运行时，此建议提供一组输入参数，与当前世界状态结合，确定新世界状态。

Response

**响应**

This section, illustrated by R4, captures the before and after values of the world state, as a Read Write set (RW-set). It's the output of a chaincode, and if the transaction is successfully validated, it will be applied to the ledger to update the world state.

本节以 R4 为例，以读写集（rw 集）的形式捕获世界状态的前后值。它是一个链码的输出，如果交易被成功验证，它将被应用到分类账以更新世界状态。

Endorsements

**赞同**

As shown in E4, this is a list of signed transaction responses from each required organization sufficient to satisfy the endorsement policy. You'll notice that, whereas only one transaction response is included in the transaction, there are multiple endorsements. That's because each endorsement effectively encodes its organization's particular transaction response - meaning that there's no need to include any transaction response that doesn't match sufficient endorsements as it will be rejected as invalid, and not update the world state.

如 e4 所示，这是每个所需组织的签名事务响应列表，足以满足背书策略。您会注意到，虽然事务中只包含一个事务响应，但有多个背书。这是因为每个认可都有效地编码了其组织的特定事务响应——这意味着不需要包括任何与足够认可不匹配的事务响应，因为它将被视为无效而拒绝，并且不更新世界状态。

That concludes the major fields of the transaction - there are others, but these are the essential ones that you need to understand to have a solid understanding of the ledger data structure.

这就总结了交易的主要领域——还有其他领域，但这些是您需要了解的基本领域，以便对分类帐数据结构有一个扎实的了解。

## 7、World State database options

## 7、世界状态数据库选项

The world state is physically implemented as a database, to provide simple and efficient storage and retrieval of ledger states. As we've seen, ledger states can have simple or complex values, and to accommodate this, the world state database implementation can vary, allowing these values to be efficiently implemented. Options for the world state database currently

include LevelDB and CouchDB.

世界状态在物理上被实现为一个数据库，以提供对分类帐状态的简单有效的存储和检索。正如我们所看到的，分类帐状态可以有简单或复杂的值，为了适应这一点，世界状态数据库的实现可能会有所不同，从而使这些值得到有效的实现。目前，世界状态数据库的选项包括 leveldb 和 couchdb。

LevelDB is the default and is particularly appropriate when ledger states are simple key-value pairs. A LevelDB database is closely co-located with a network node – it is embedded within the same operating system process.

LEVELDB 是默认值，当分类帐状态是简单的键值对时尤其适用。一个 LevelDB 数据库与一个网络节点紧密地位于同一个操作系统进程中。

CouchDB is a particularly appropriate choice when ledger states are structured as JSON documents because CouchDB supports the rich queries and update of richer data types often found in business transactions. Implementation-wise, CouchDB runs in a separate operating system process, but there is still a 1:1 relation between a network node and a CouchDB instance. All of this is invisible to chaincode. See CouchDB as the StateDatabase for more information on CouchDB.

当分类帐状态结构化为 JSON 文档时，CouchDB 是一个特别合适的选择，因为 CouchDB 支持丰富的查询和更新业务事务中经常出现的更丰富的数据类型。在实现方面，CouchDB 在单独的操作系统进程中运行，但在网络节点和 CouchDB 实例之间仍然存在 1:1 的关系。所有这些对于链码来说都是不可见的。有关 couchdb 的详细信息，请参阅 couchdb 作为状态数据库。

In LevelDB and CouchDB, we see an important aspect of Hyperledger Fabric – it is pluggable. The world state database could be a relational data store, or a graph store, or a temporal database. This provides great flexibility in the types of ledger states that can be efficiently accessed, allowing Hyperledger Fabric to address many different types of problems.

在 LevelDB 和 CouchDB 中，我们看到了 Hyperledger 结构的一个重要方面——它是可插拔的。世界状态数据库可以是关系数据存储、图形存储或临时数据库。这在可以有效访问的分类帐状态类型中提供了很大的灵活性，允许 Hyperledger 结构处理许多不同类型的问题。

## 8、Example Ledger: fabcar

## 8、分类帐示例：Fabcar

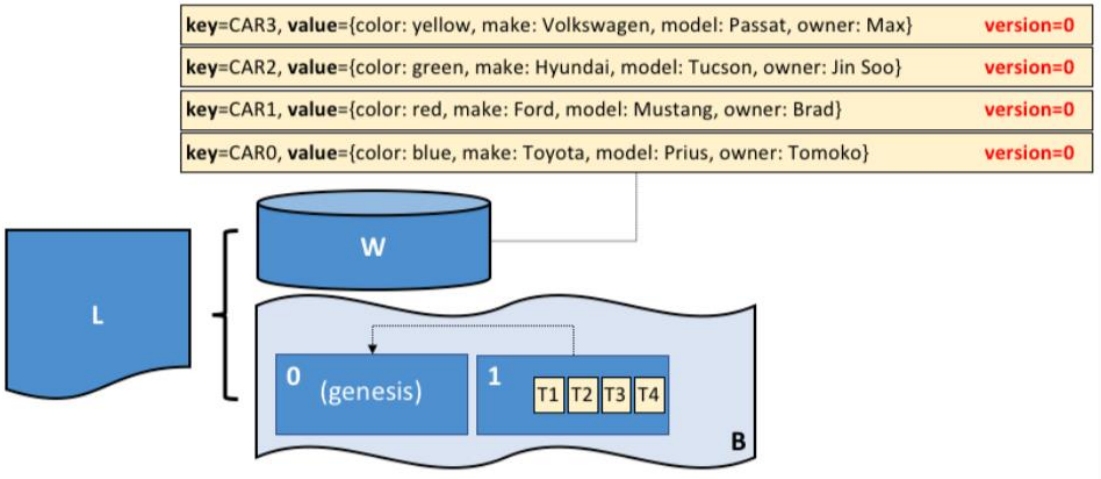
As we end this topic on the ledger, let’s have a look at a sample ledger. If you’ve run the fabcar sample application, then you’ve created this ledger.

当我们在分类帐上结束这个主题时，让我们看一看示例分类帐。如果您已经运行了 Fabcar 示例应用程序，那么您已经创建了这个分类账。

The fabcar sample app creates a set of 10 cars, of different color, make, model and owner. Here’s what the ledger looks like after the first four cars have been created.

Fabcar 示例应用程序创建了一组 10 辆不同颜色、品牌、型号和所有者的汽车。这

是创建前四辆汽车后的分类账。




The visual vocabulary expressed in facts is as follows: The ledger L, comprises a world state, W and a blockchain, B. W contains four states with keys: CAR1, CAR2, CAR3 and CAR4. B contains two blocks, 0 and 1. Block 1 contains four transactions: T1, T2, T3, T4.

用事实表示的视觉词汇如下：分类帐 L，由一个世界国家、W 和一个区块链组成，B.W 包含四个国家，其键为：car1、car2、car3 和 car4。B 包含两个块，0 和 1。块 1 包含四个事务：T1、T2、T3、T4。

We can see that the ledger world state contains states that correspond to CAR0, CAR1, CAR2 and CAR3. CAR0 has a value which indicates that it is a blue Toyota Prius, owned by Tomoko, and we can see similar states and values for the other cars. Moreover, we can see that all car states are at version number 0, indicating that this is their starting version number - they have not been updated since they were created.

我们可以看到，分类帐世界状态包含对应于 car0、car1、car2 和 car3 的状态。car0 的值表示它是 Tomoko 拥有的蓝色丰田普锐斯，我们可以看到其他汽车的类似状态和值。此外，我们可以看到所有汽车状态的版本号都是 0，这表明这是它们的初始版本号——它们自创建以来就没有更新过。

We can also see that the ledger blockchain contains two blocks. Block 0 is the genesis block, though it does not contain any transactions that relate to cars. Block 1 however, contains transactions T1, T2, T3, T4 and these correspond to transactions that created the initial states for CAR0 to CAR3 in the world state. We can see that block 1 is linked to block 0.

我们还可以看到分类账区块链包含两个区块。0 块是 Genesis 块，虽然它不包含任何与汽车相关的事务。但是，块 1 包含事务 T1、T2、T3、T4，这些事务对应于在世界状态下为 car0 到 car3 创建初始状态的事务。我们可以看到块 1 链接到块 0。

We have not shown the other fields in the blocks or transactions, specifically headers and hashes. If you're interested in the precise details of these, you will find a dedicated reference topic elsewhere in the documentation. It gives you a fully worked example of an entire block with its transactions in glorious detail - but for now, you have achieved a solid conceptual understanding of a Hyperledger Fabric ledger. Well done!

我们没有在块或事务中显示其他字段，特别是头和散列。如果您对这些内容的精确细节感兴趣，可以在文档的其他地方找到一个专门的参考主题。它为您提供了一个完整的例子，展示了一个完整的模块，其中的事务处理非常详细，但是现在，您已经对一个超级账本结构分类账有了一个坚实的概念理解。做得好！

### 9、More information

### 9、更多信息

See the Transaction Flow, Read-Write set semantics and CouchDB as the StateDatabase topics for a deeper dive on transaction flow, concurrency control, and the world state database.

请参阅事务流、读写集语义和 couchdb 作为状态数据库主题，以深入了解事务流、并发控制和世界状态数据库。

## 十、Use Cases

## 十、用例

The Hyperledger Requirements WG is documenting a number of blockchain use cases and maintaining an inventory here.

Hyperledger Requirements 工作组正在记录大量区块链用例，并在此处维护库存。