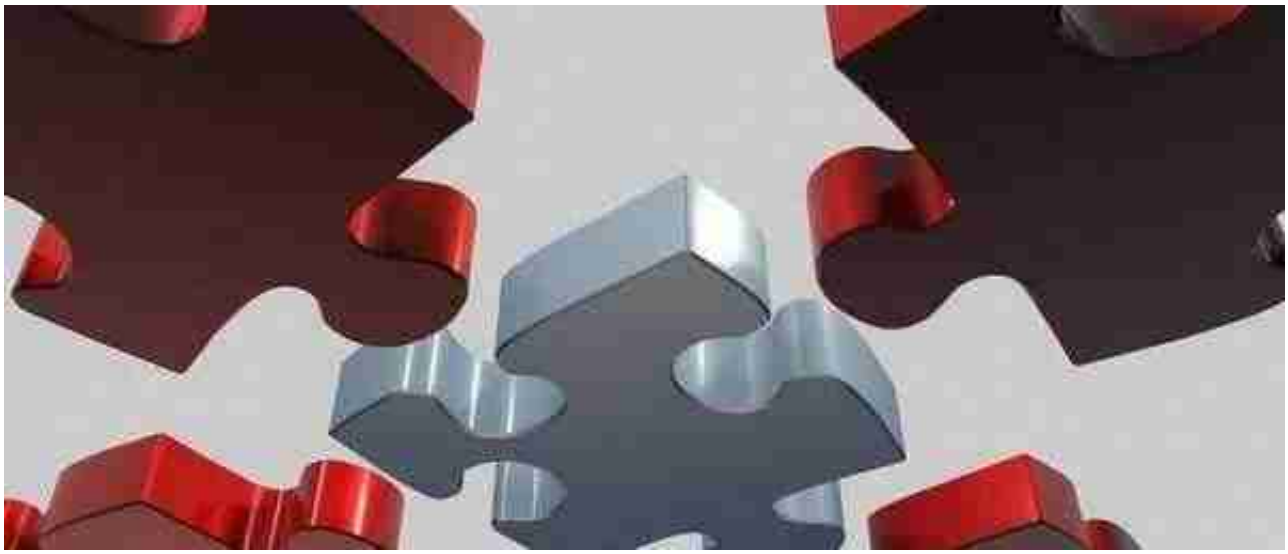
		比特币	以太坊	Hyperledger Fabric
应用层		比特币交易	Dapp/以太币交易	企业级区块链应用
智能合约层	编程语言	Script	Solidity/Serpent	Go/Java
	沙盒环境		EVM	Dockr
数据层	数据结构	Merkel树/区块链表	Merkle Patricia树/区块链表	Merkle Patricia树/区块链表
	数据模型	基于交易的模型	基于账户的模型	基于账户的模型
	区块存储	文件存储	LevelDB	文件存储
共识层		PoW	PoW/PoS	PBFT/SBFT

网络层

网络层的主要目的是实现区块链网络节点之间的信息交互。区块链的本质是一个点对点(P2P)网络，每一个节点既能够接收信息，也能够生产信息，节点之间 通过维护一个共同的区块链来保持通信。在区块链的网络中，每一个节点都可以创造出新的区块，新区块被创造出以后，
会通过广播的形式通知其他的节点，而其他节点反过来会对这个节点进行验证。当区块链网络中超过 51%
的用户对其验证通过以后，这个新的区块就会被添加到主链上



数据层

数据层是最底层的技术，主要的功能为数据存储、账户和交易的实现与安全。数据存储主要基于 Merkle 树，通过区块的方式和链式结构实现，大多以 KV 数据库的方式实现持久化，如比特币和以太坊采用的 LevelDB。基于数字签名、散列函数、非对称加密技术等多种密码学算法和技术，以及账
户和交易的实现与安全功能，保证了交易能够在去中心化的情况下安全进行。

设计区块链系统的技术人员们首先建立的起始节点，被称作是“创世区块”，
之后在同样的规则之下，创建规格相同的区块，通过一个链式结构依次相连组成一条主链。随着运行时间的增加，新的区块通过验证后，被不断添加到主链上，主链会不断延长。

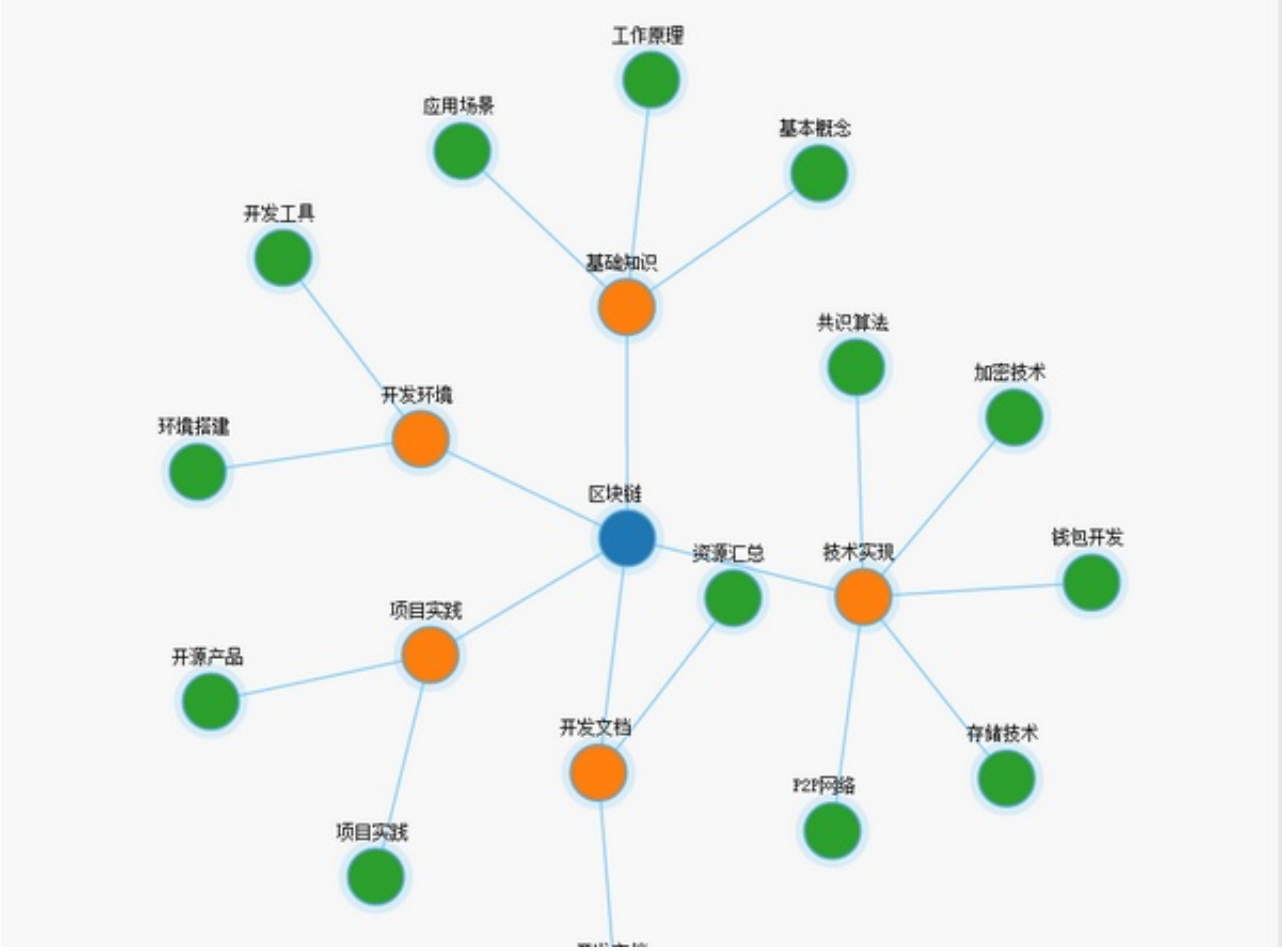
每一个区块中同时也包含了许多技术，如时间戳技术，它的作用在于确保每一个区块都可以按时间的顺序相连接，比如散列函数，它是一种将任意长度的消息通过散列算法压缩到某一固定长度的消息摘要的函数，它主要用于信息安全领域中加密算法、文件检验、数字签名和鉴权协议等。

合约层

所谓合约层主要是指各种脚本代码、算法机制及智能合约等。智能合约是运行在区块链上的一段无须干预即可自动执行的代码，EVM
是智能合约运行的虚拟机，
人类通过智能合约，无须任何中介干预即可实现资产的转移，同时也可以开发出一些有价值的去中心化应用。以比特币为例，它是一种可编程的数字货币，合约层封

装的脚本中规定了比特 币的交易方式和交易过程中所涉及的各种细节。

基于智能合约还可以构建区块链应用，不需要从零学习区块链技术就可以方便地开发自己的区块链应用(DAPP)。如基于以太坊公链，开发者可以使用 Solidity 语言开发智能合约，构建去中心化应用 ;基于 EOS，开发者可以使用 C++ 语言，编写自己的智能合约。



基础知识

区块链是新技术，与之相关的是其背后大量的新概念、新理论。这些知识，虽然不直接体现在编码里，但却是理解区块链，掌握区块链技术的基本知识。所以，理当成为区块链技术不可或缺的一部分。这部分从基本概念入手，到工作原理的描述，就能够把区块链基础知识全部覆盖。

技术实现

区块链是一项技术，但从上面的分析可以看出，它应该是一种架构应用，架构的实

现理当是我们知识库的核心。正如大家看到的，任何一款区块链产品，协议层必须包括点对点网络、加密签名、数据存储、分布式算法等4个部分，应用层也必然要提供钱包、客户端浏览器等基础应用。所以，把这部分独立出来，也是合情合理。

在扩展层的部分，区块链技术可以对接各种应用，比如：金融、物联网、网络安全、版权保护、电子商务等等，现有的很多技术都可以用在这里。只不过，如何与区块链结合，如何实现跨行业使用，自然是这部分内容研究的课题。所以，这里所罗列或涉及到的技术，理应归为技术实现的一个重要部分。

开发环境

区块链是多项技术的组合，有其自身的复杂性，个别应用对开发环境依赖较大，开发工具与环境搭建，是让开发者快速上手的重要内容，据说，短短数年，全球区块链产品已经有几千个，其中不乏创新应用。有些优秀的开源产品和项目实践，是最好的学习研究资料。

编程实现

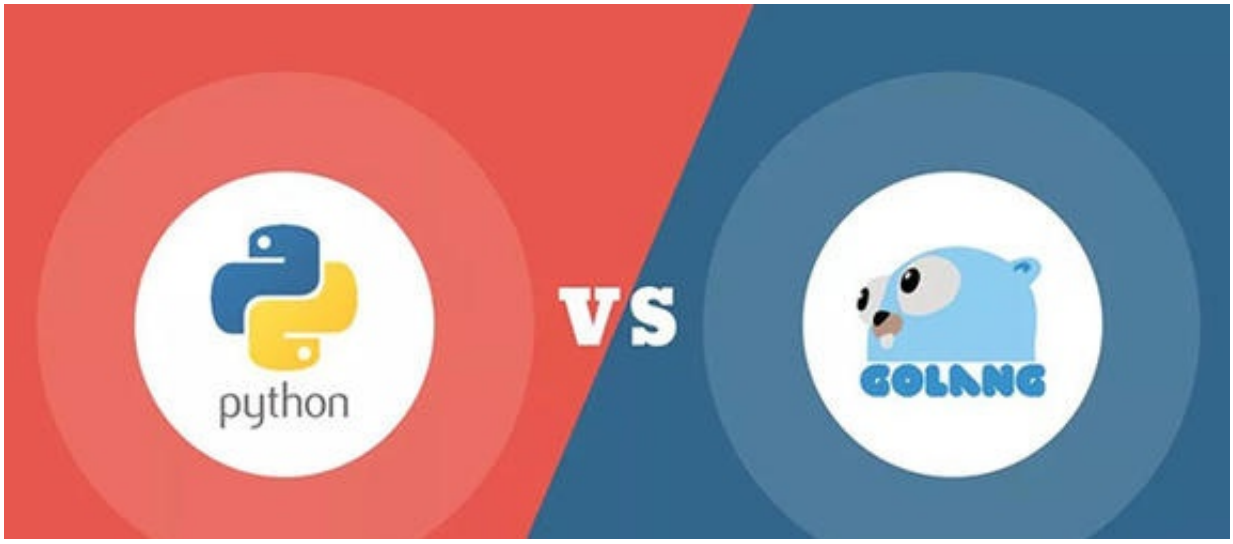
C/C++实现

这两个语言是无法逾越的，任何开发遇到瓶颈，基本上都会找到它们，自然应该排在第一位要介绍的。同时，区块链技术的鼻祖，比特币(协议层) 就是用C++语言开发的，而且目前为止，没有比比特币更加成功的区块链产品。所以，无论你使用什么语言开发，在正式进入这个行业的过程中，都应该先研究研究比特币。

它提供了强大的协作机制，为数字出版、版权保护提供了便利；扩展了侧链功能，可以基于它开发任何去中心化的应用，从而为专业作者、博客爱好者和开发者提供很多方便。《Nodejs开发加密货币》这本书完整分享了它的源码，从区块链基础概念到代码实现，从基本原理到开发设计思路，都做了比较详细的探索，目前为止，从协议层面深入代码讲解区块链技术实现的书籍极少，这算作一本。

Python

如果是Python语言爱好者，我建议研究研究以太坊（Ethereum）的Python实现。尽管因为The Dao事件闹得沸沸扬扬，但从技术实现的角度来说，仍然值得参考学习。以太坊官方定位为一种开发管理分布式应用的平台，主攻方向就是“智能合约”，并为其定制了一种编程语言Solidity。以太坊的核心是以太坊虚拟机（EVM），允许用户按照自己的意愿创建操作。以太坊给出了Go、Java、Python等多语言的实现。



Fabric的开发环境建立在VirtualBox虚拟机上，部署环境可以自建网络，也可以直接部署在BlueMix上，部署方式可docker化，支持用Go和JavaScript开发智能合约。它采用PBFT分布式算法，网络编程方面用gRPC来做P2P通讯，使用 Protocol Buffer来序列化要传递的数据结构。在架构设计上，Fabric可能与比特币等区块链产品有所不同，但是上述基本组成部分还是不可或缺的。