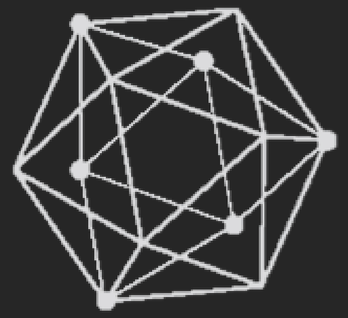
Work done at

IBM



github.com/hyperledger/fabric

Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains

Eurosys 2018

CCF B

目　　录

介绍

Fabric 架构

Fabric 组件

实验评估

应用案例

1 Fabric 背景介绍

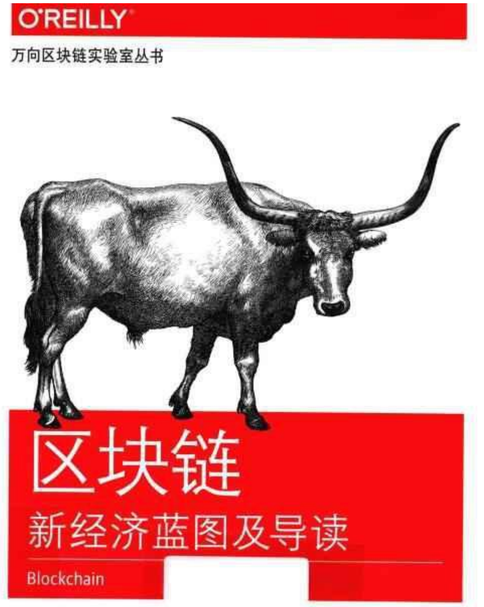
HYPERLEDGER

THE LINUX FOUNDATION

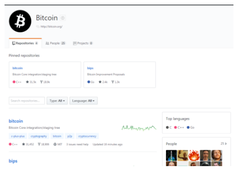
ethereum

ripple

HYPERLEDGER



2008.10 2009.1 2014.10

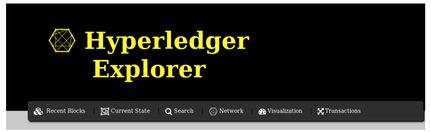


1 Fabric 背景介绍

HYPERLEDGER

HYPERLEDGER FABRIC

HYPERLEDGER SAWTOOTH



HYPERLEDGER IROHA

HYPERLEDGER CELLO

HYPERLEDGER COMPOSER

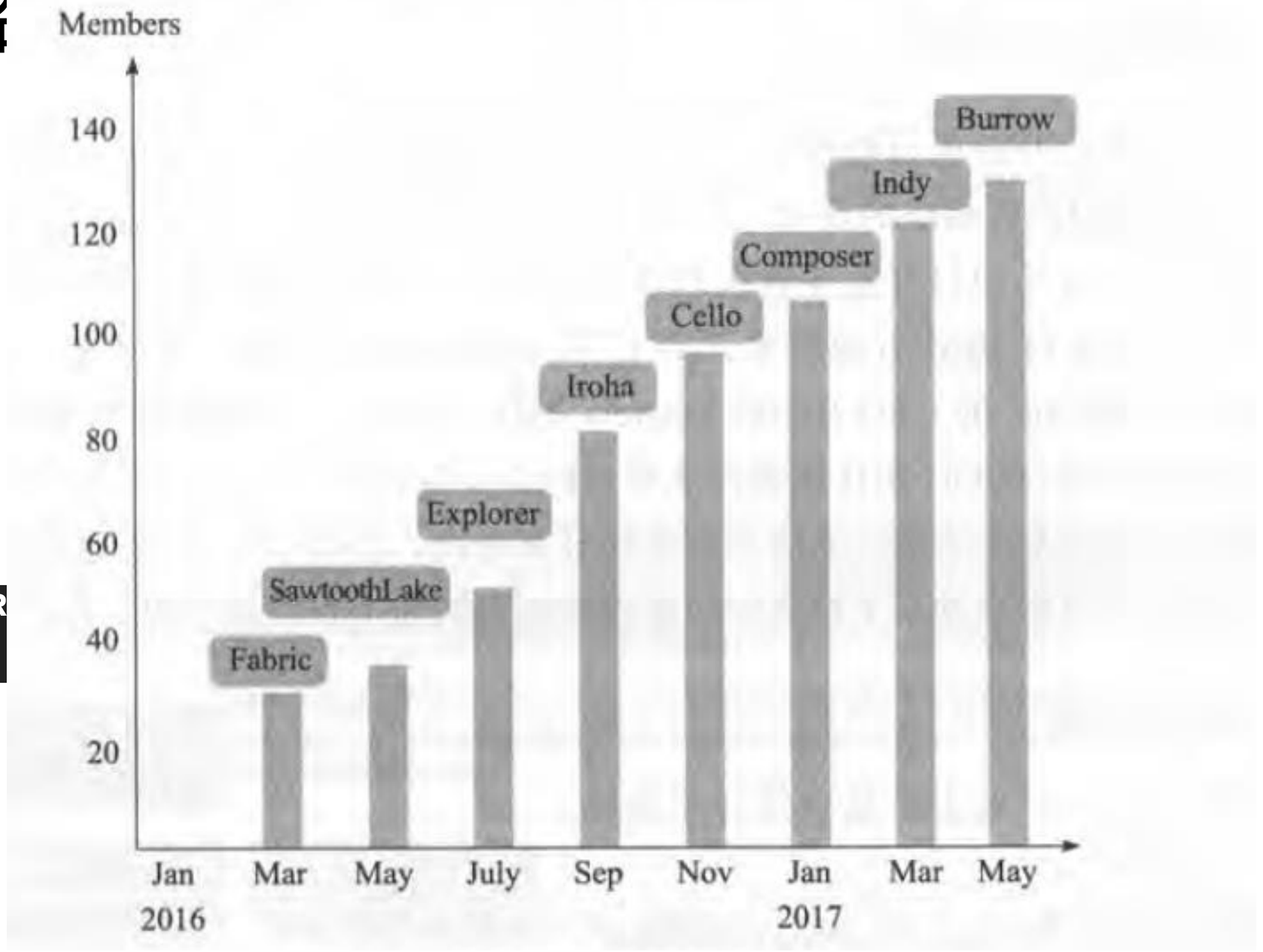
HYPERLEDGER INDY

1 Fabric 背景介绍

HYPERLEDGER

HYPERLEDGER FABRIC

http://github.com/hyperleder/fabric

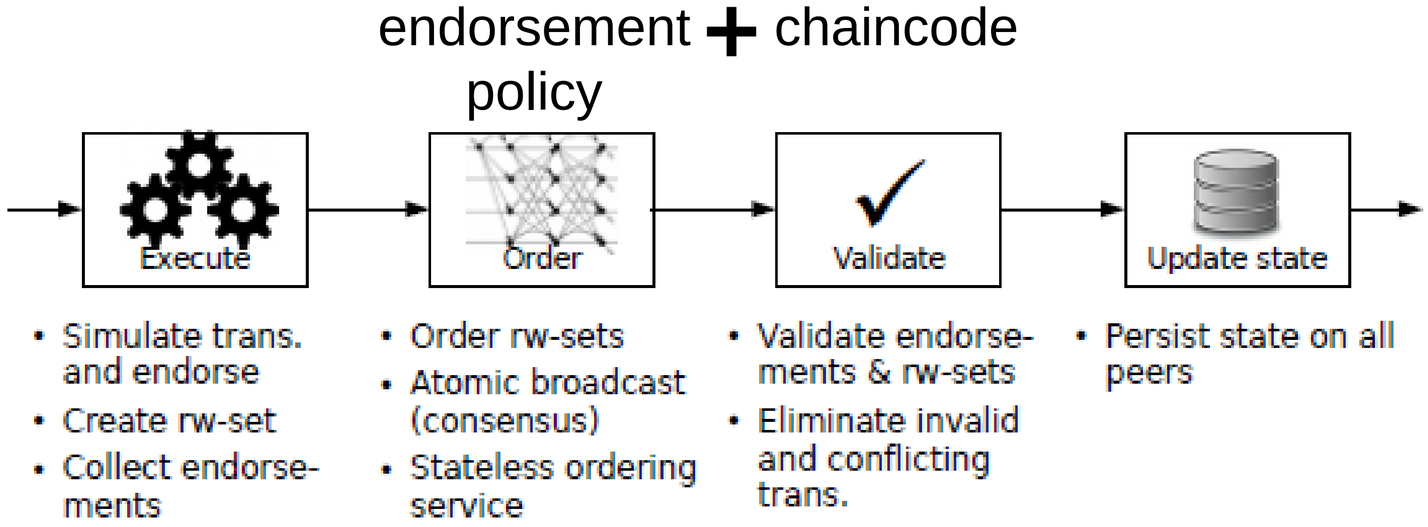


1 Fabric 背景介绍

Prior permissioned blockchains suffer from many limitations:

1. Consensus is hard-coded within the platform, which contradicts the well-established understanding that there is no “one-size-fits-all” (BFT) consensus protocol [52];
2. The trust model of transaction validation is determined by the consensus protocol and cannot be adapted to the requirements of the smart contract;
3. Smart contracts must be written in a fixed, non-standard, or domain-specific language, which hinders wide-spread adoption and may lead to programming errors;
4. The sequential execution of all transactions by all peers limits performance, and complex measures are needed to prevent denial-of-service attacks against the platform originating from untrusted contracts (such as accounting for runtime with “gas” in Ethereum);
5. Transactions must be deterministic, which can be difficult to ensure programmatically;
6. Every smart contract runs on all peers, which is at odds with confidentiality, and prohibits the dissemination of contract code and state to a subset of peers.

2 Fabric 架构



A distributed application

Figure 2: Execute-order-validate architecture of Fabric (*rw-set* means a readset and writeset as explained in Sec. 3.2).

2 Fabric 架构

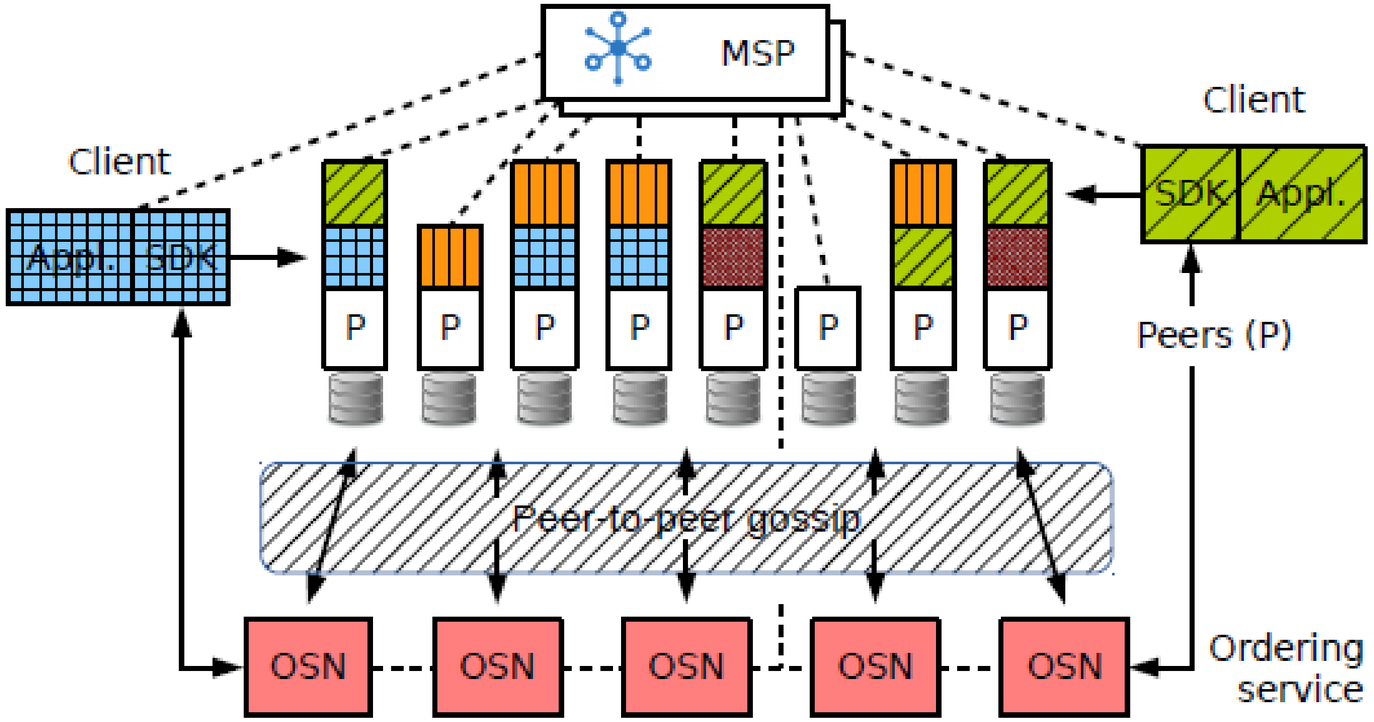


Figure 3: A Fabric network with federated MSPs and running multiple (differently shaded and colored) chaincodes, selectively installed on peers according to policy.

Fabric 架构

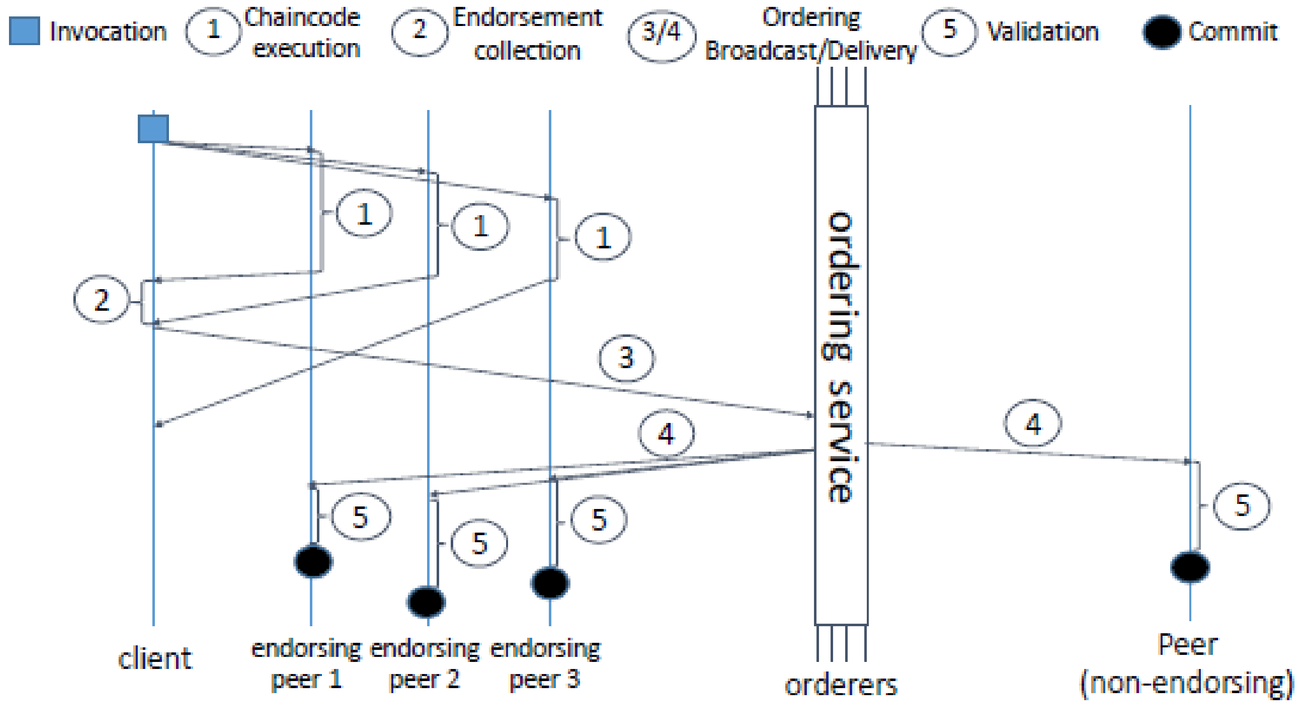


Figure 4: Fabric high level transaction flow.

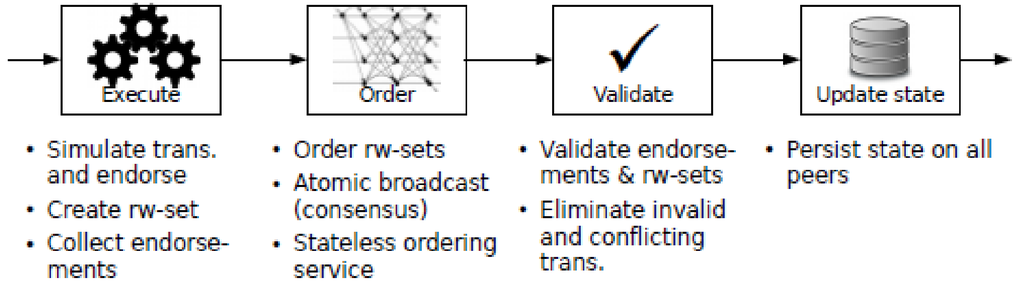


Figure 2: Execute-order-validate architecture of Fabric (*rw-set* means a readset and writeset as explained in Sec. 3.2).

Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

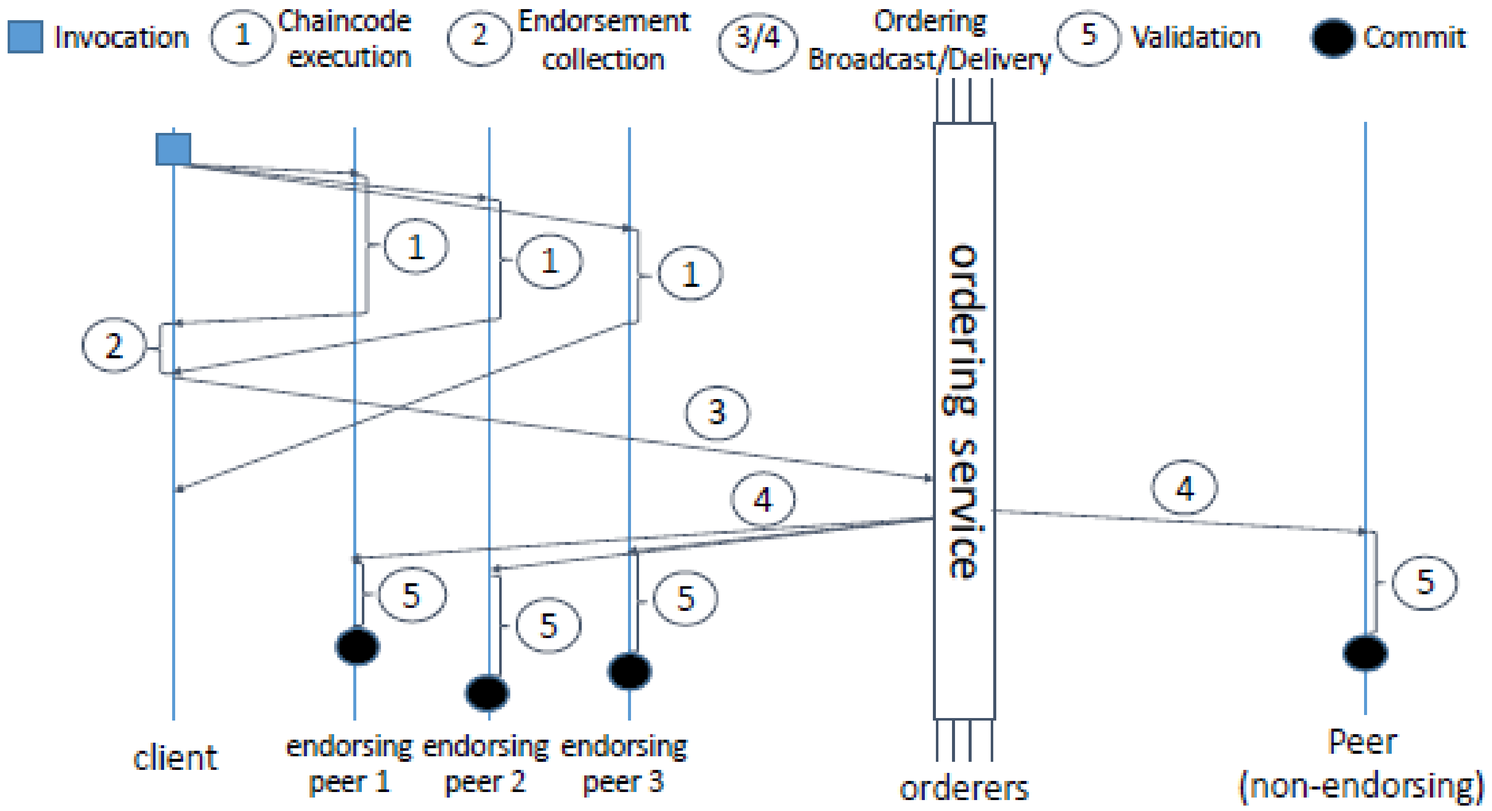


Figure 4: Fabric high level transaction flow.

Fabric 组件

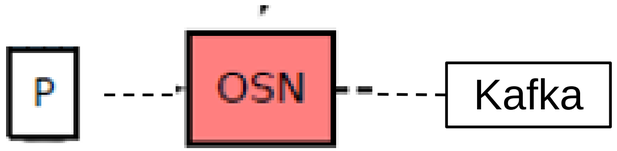
- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

- identities of all nodes in the system (clients, peers, and OSNs) and is responsible for issuing node credentials that are used for authentication and authorization.;
- The default MSP implementation in Fabric handles standard PKI methods for authentication based on digital signatures and can accommodate commercial certification authorities (CAs).;
- offline mode and online mode;

Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

- Atomic broadcast for establishing order on transactions, implementing the broadcast and deliver calls (Sec. 3.3).
- Reconfiguration of a channel, when its members modify the channel by broadcasting a configuration update transaction (Sec. 4.6).
- Optionally, access control, in those configurations where the ordering service acts as a trusted entity, restricting broadcasting of transactions and receiving of blocks to specified clients and peers;



Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

- Push
- pull;

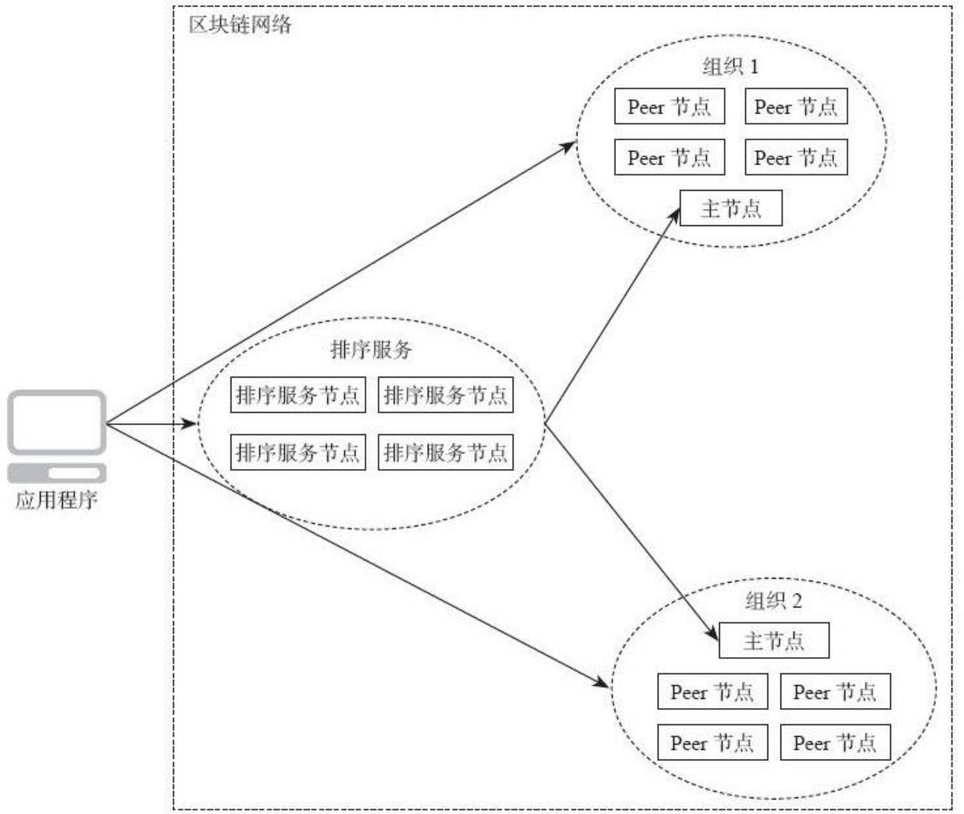


图4-1　基于Gossip的通信路径

Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

- Consist of a block store and a peer transaction manager
- readset and writeset;

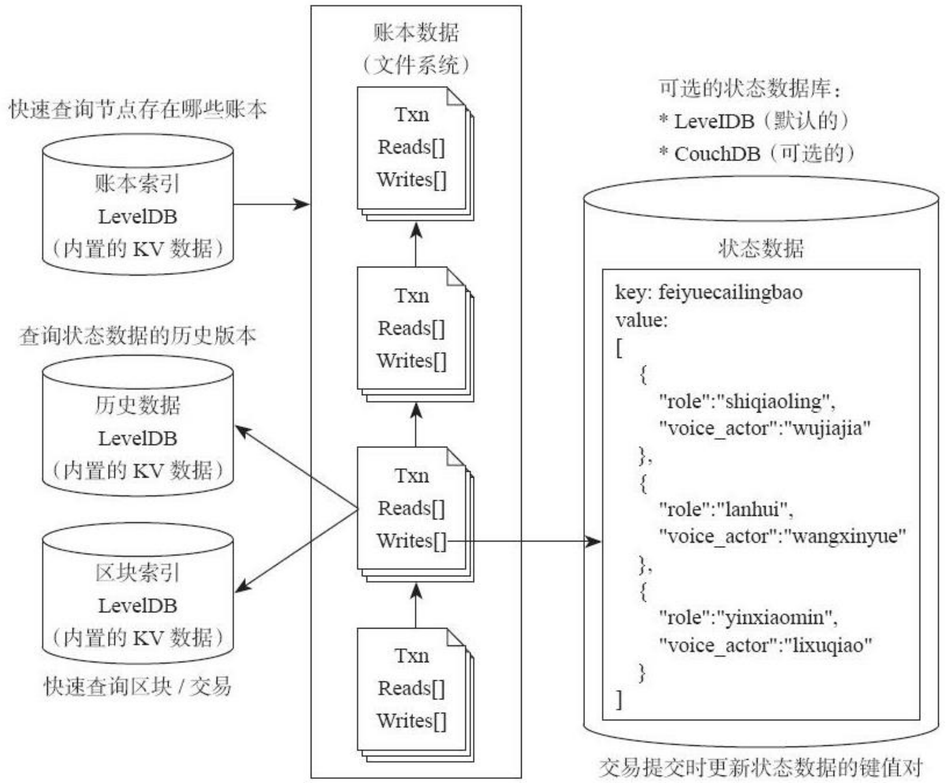


图5-1　分布式账本存储

Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

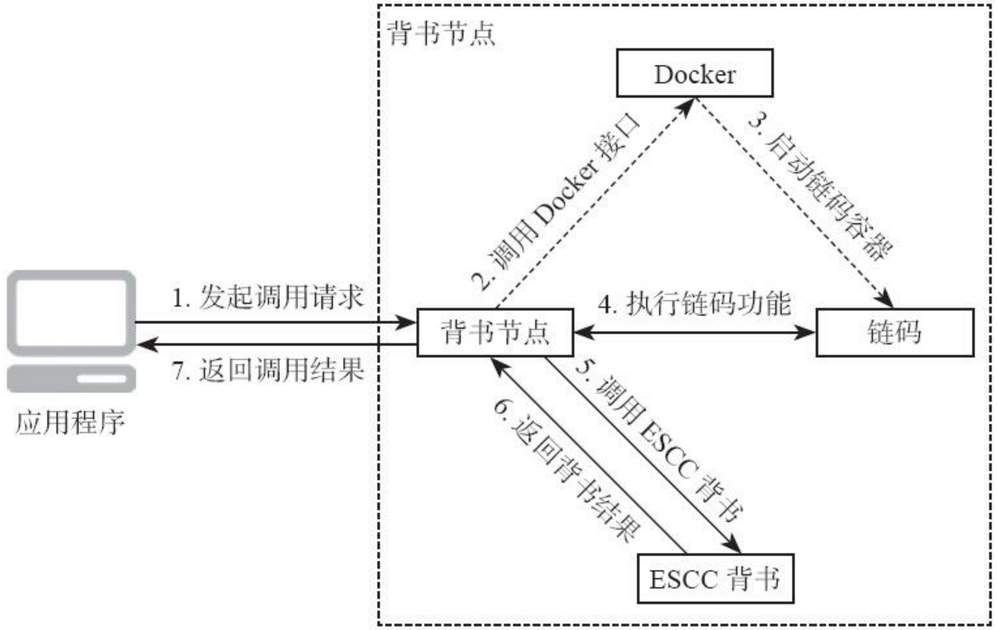


图9-1　应用程序和链码的交互流程

Fabric 组件

- Membership Service;
- Ordering Service;
- Peer Gossip;
- Ledger;
- Chaincode Execution;
- Configuration and System Chaincodes;

- Fabric is customized through channel configuration and through special chaincodes, known as system chaincodes.
- The channel configuration includes: (1) definitions of the MSPs for the participating nodes, (2) the network addresses of the OSNs, (3) shared configuration for the consensus implementation and the ordering service, such as batch size and timeouts, (4) rules governing access to the ordering service operations (broadcast, and deliver), and (5) rules governing how each part the channel configuration may be modified.
- The application chaincodes are deployed with a reference to an endorsement system chaincode (ESCC) and to a validation system chaincode (VSCC).

实验评估

- Fabric Coin (Fabcoin);
- Experiments
  - Setup;
  - Methodology;

- (1) Fabric version v1.1.0-preview2
- (2) nodes are hosted in a single IBM Cloud (SoftLayer) data center (DC) as dedicated VMs interconnected with 1Gbps (nominal) networking,
- (3) all nodes are 2.0 GHz 16-vCPU VMs running Ubuntu with 8GB of RAM and SSDs as local disks,
- (4) a single-channel ordering service runs a typical Kafka orderer setup with 3 ZooKeeper nodes, 4 Kafka brokers and 3 Fabric orderers, all on distinct VMs,
- (5) there are 5 peers in total, all belonging to different organizations (orgs) and all being Fabcoin endorsers, and
- (6) signatures use the default 256-bit ECDSA scheme.

实验评估

- Fabric Coin (Fabcoin);

- Experiments
  - Setup;
  - Methodology;

- Fabcoin mint
- Fabcoin spend

实验评估

- Experiment 1
  - Experiment 1：Choosing the block size
  - Experiment 2：Impact of peer CPU
  - Experiment 3：SSD vs. RAM disk
  - Experiment 4：Scalability on LAN
  - Experiment 5：Scalability over two DCs and impact of gossip
  - Experiment 6：Performance over multiple data centers (WAN).

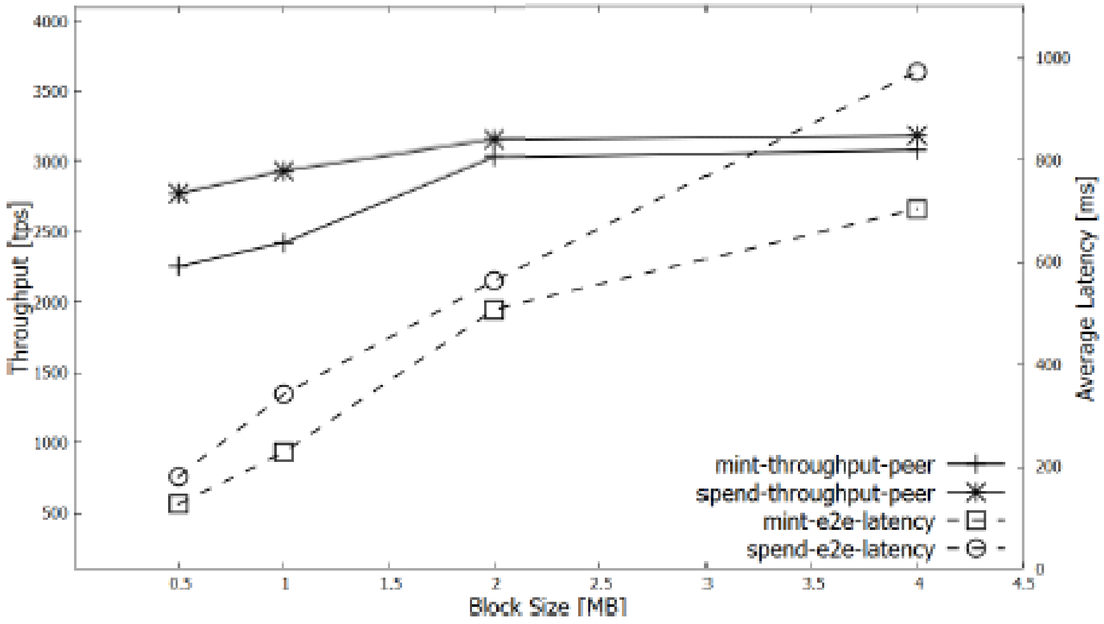
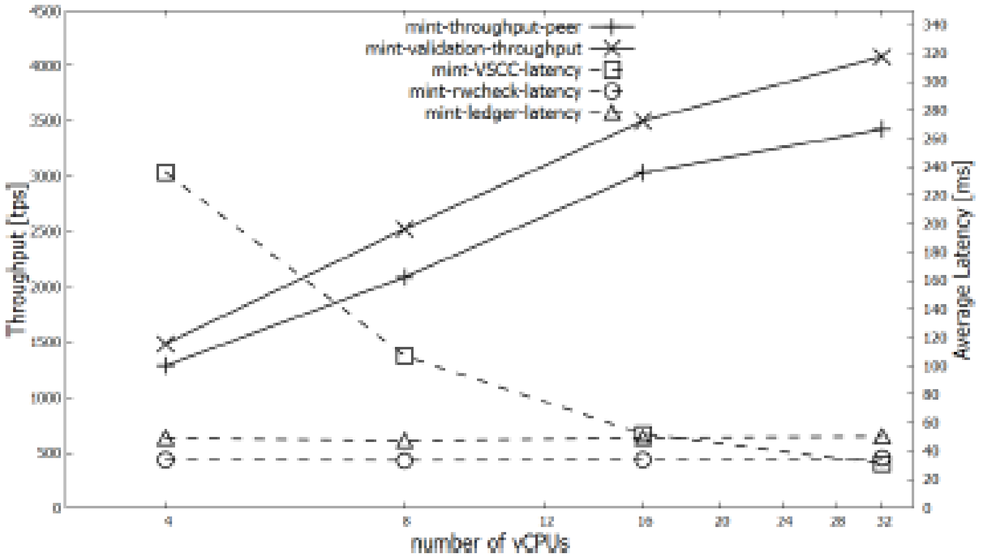


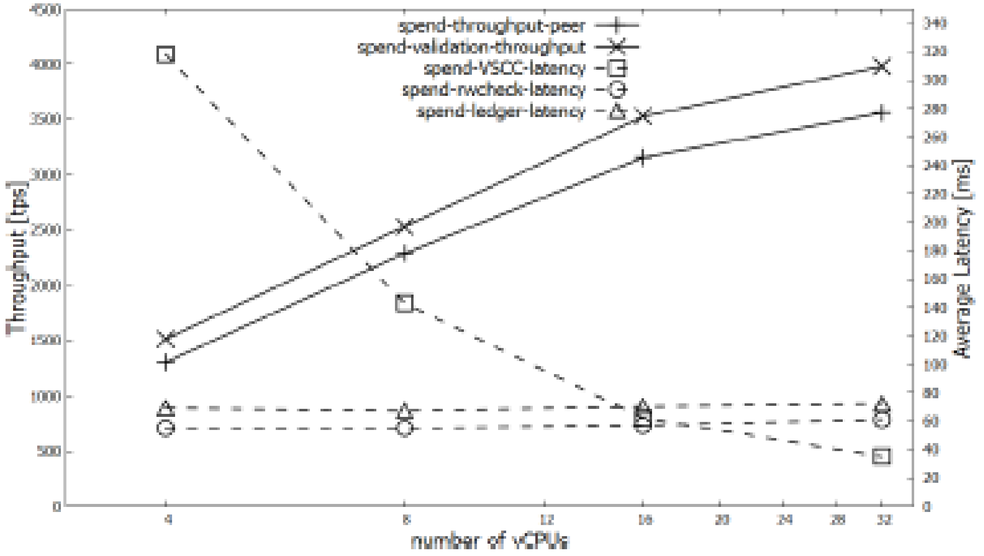
Figure 6: Impact of block size on throughput and latency.

实验评估

- Experiment 1
  - Experiment 1：Choosing the block size
  - Experiment 2：Impact of peer CPU
  - Experiment 3：SSD vs. RAM disk
  - Experiment 4：Scalability on LAN
  - Experiment 5：Scalability over two DCs and impact of gossip
  - Experiment 6：Performance over multiple data centers (WAN).



(a) Blocks containing only MINT transactions.



(b) Blocks containing only SPEND transactions.

Figure 7: Impact of peer CPU on end-to-end throughput, validation throughput and block validation latency.

实验评估

- Experiment 1
  - Experiment 1：Choosing the block size
  - Experiment 2：Impact of peer CPU
  - Experiment 3：SSD vs. RAM disk
  - Experiment 4：Scalability on LAN
  - Experiment 5：Scalability over two DCs and impact of gossip
  - Experiment 6：Performance over multiple data centers (WAN).

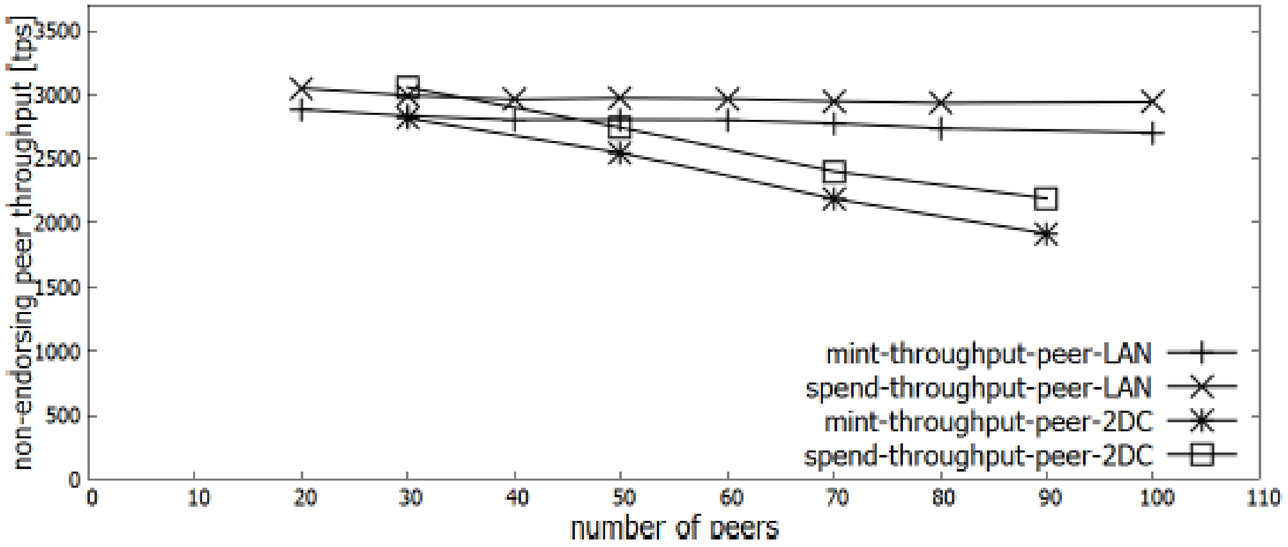


Figure 8: Impact of varying number of peers on non-endorsing peer throughput.

实验评估

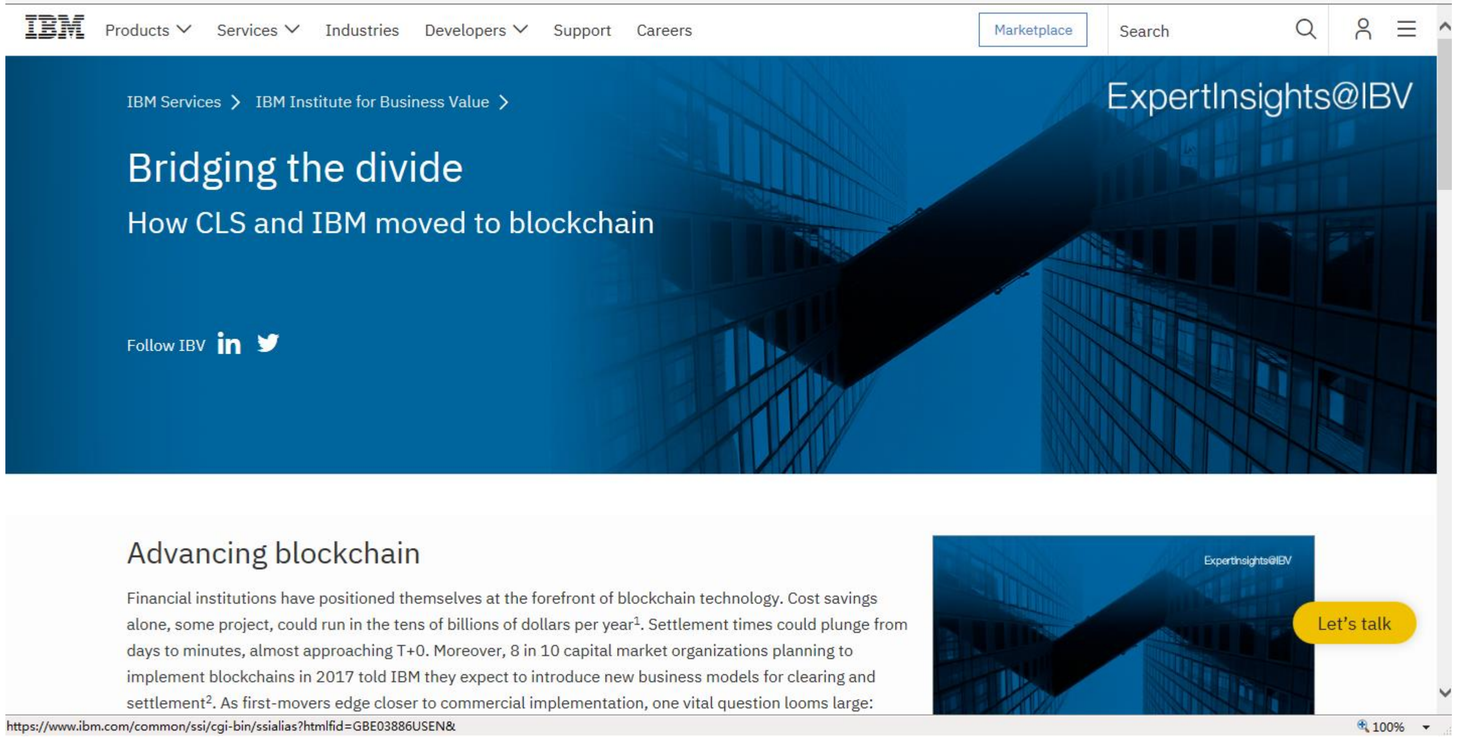
- Experiment 1
  - Experiment 1：Choosing the block size
  - Experiment 2：Impact of peer CPU
  - Experiment 3：SSD vs. RAM disk
  - Experiment 4：Scalability on LAN
  - Experiment 5：Scalability over two DCs and impact of gossip
  - Experiment 6：Performance over multiple data centers (WAN).

	HK	ML	SD	OS
netperf to TK [Mbps]	240	98	108	54
peak MINT / SPEND throughput [tps] (without gossip)	1914 / 2048	1914 / 2048	1914 / 2048	1389 / 1838
peak MINT / SPEND throughput [tps] (with gossip)	2553 / 2762	2558 / 2763	2271 / 2409	1484 / 2013

Table 2: Experiment with 100 peers across 5 data centers.

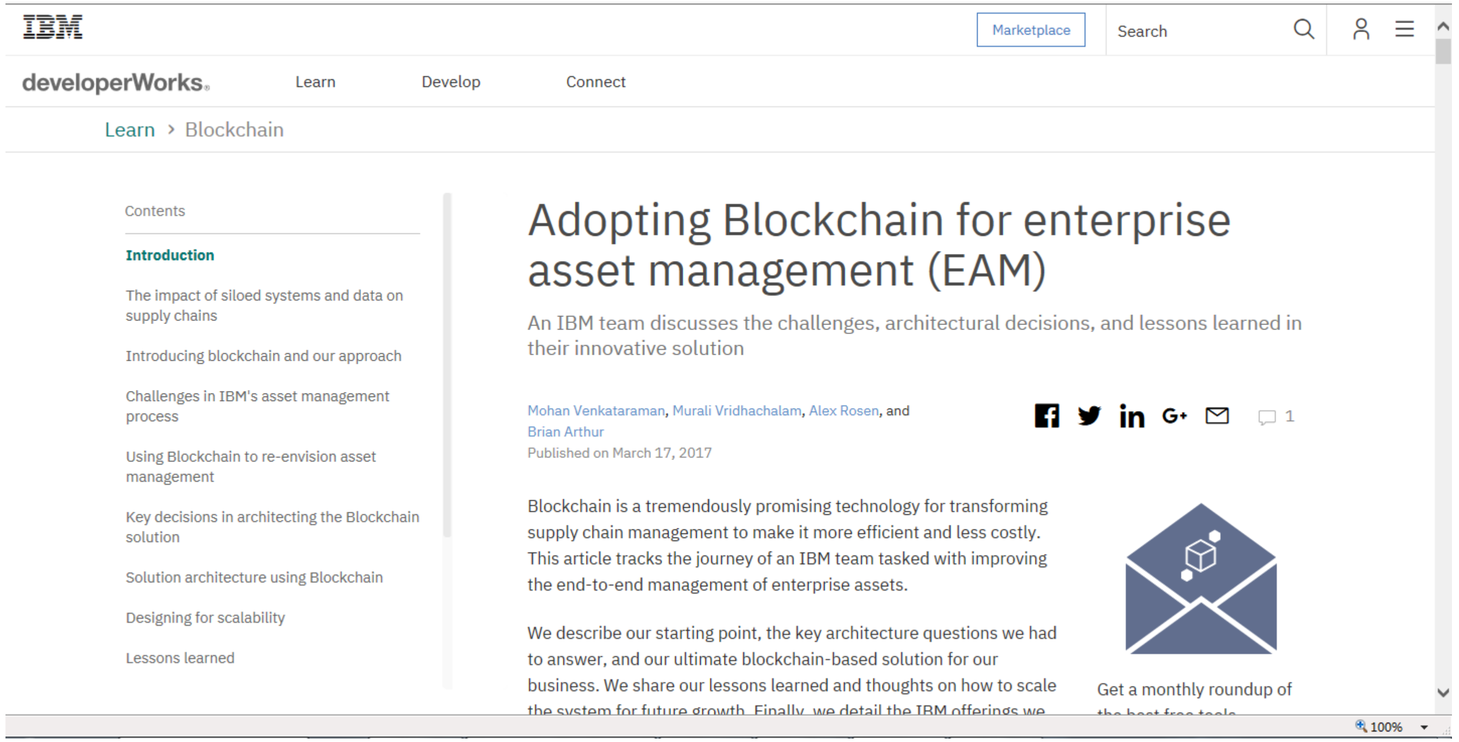
应用案例

- Foreign exchange (FX) netting
- Enterprise asset management (EAM).
- Global cross-currency payments



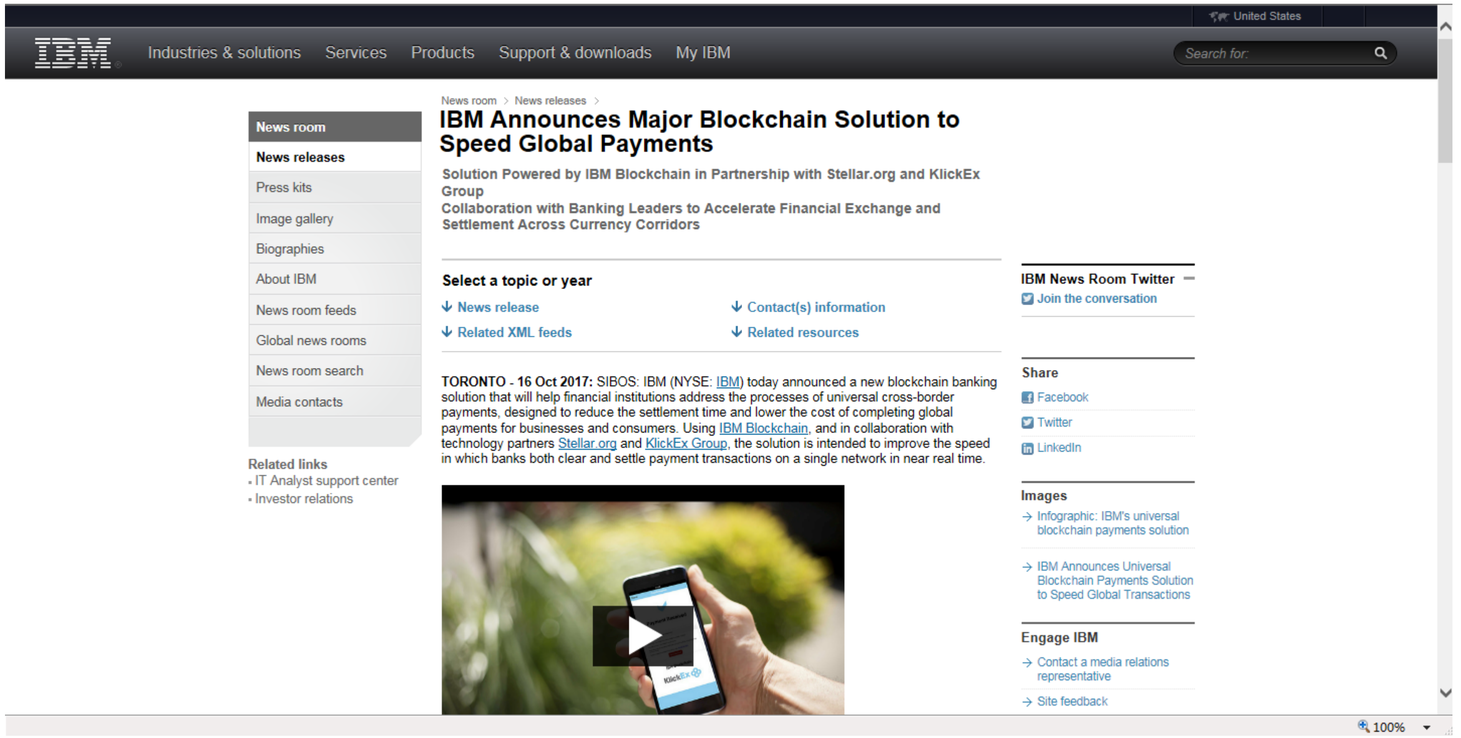
应用案例

- Foreign exchange (FX) netting
- Enterprise asset management (EAM).
- Global cross-currency payments



应用案例

- Foreign exchange (FX) netting
- Enterprise asset management (EAM).
- Global cross-currency payments



Thank you
Q & A