

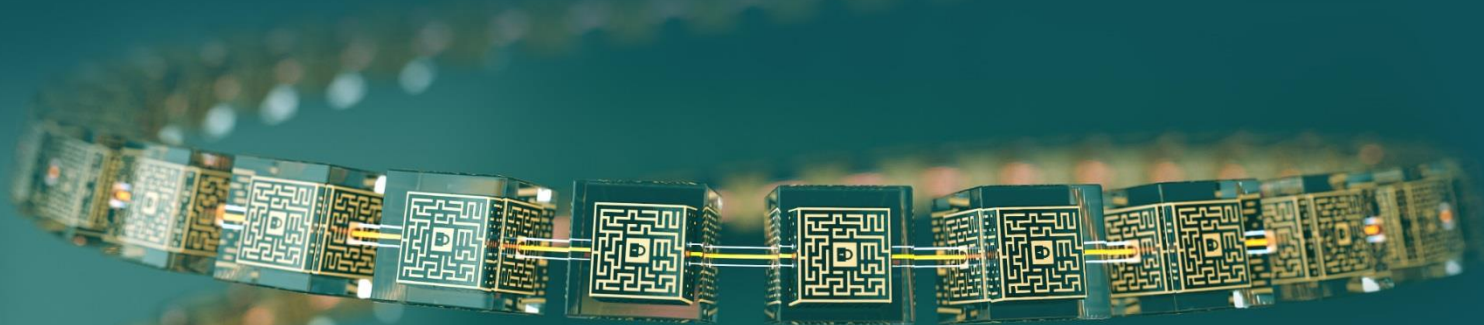
J1 : Blockchain 세미나

블록체인으로 변화될 미래의 모습

블록체인 기술의 이해와 응용

V4.0 June 27

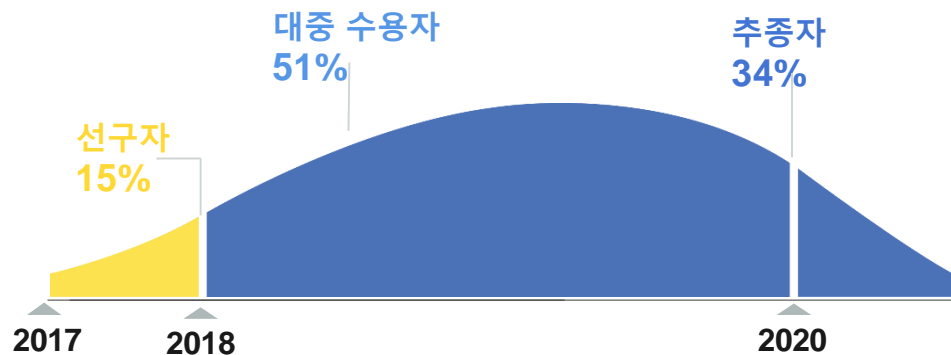
© 2017 IBM Corporation



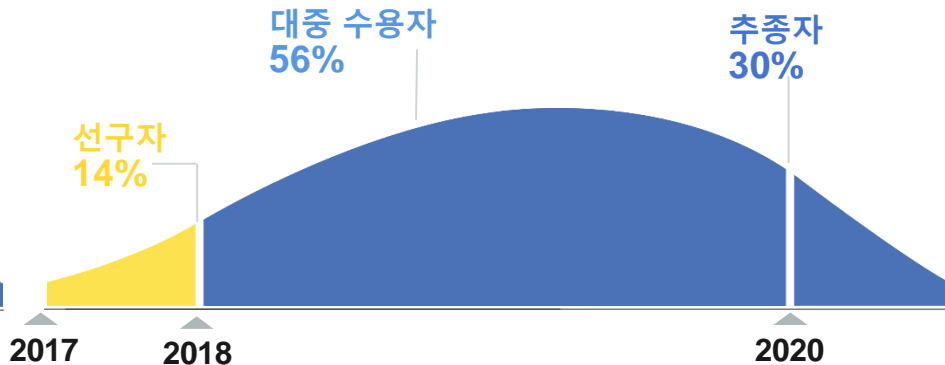
산업별 블록체인 활용 전망

2017년에 대규모로 상용솔루션 구축을 예상하는 선구자 그룹이 블록체인 적용을 선도할 것으로 예상됨

은행 산업



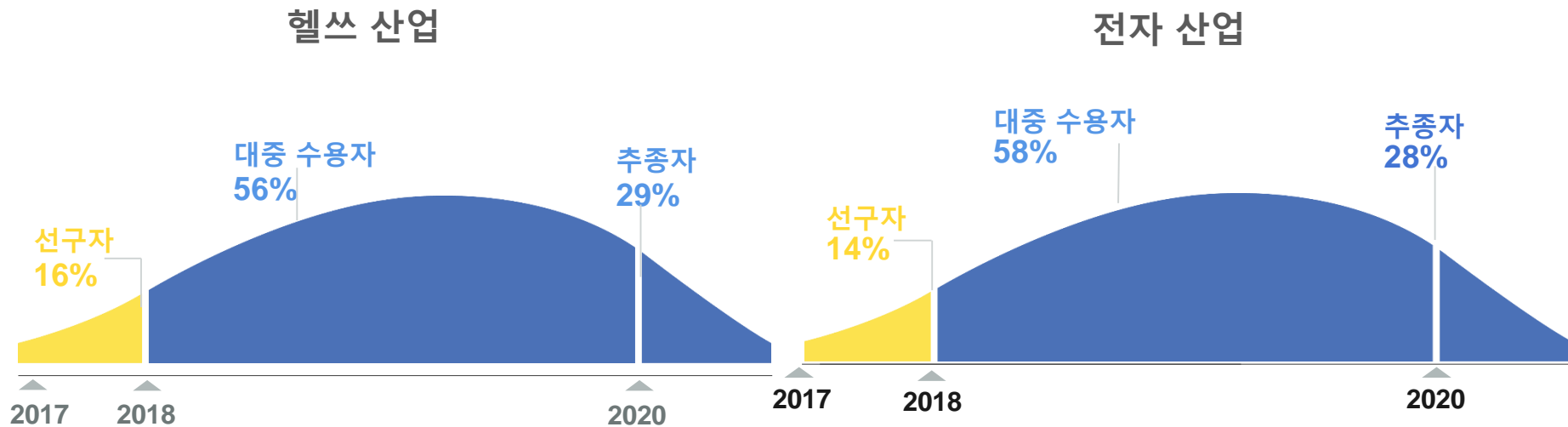
자본시장 산업



Source: IBM Institute for Business Value analysis

산업별 블록체인 활용 전망

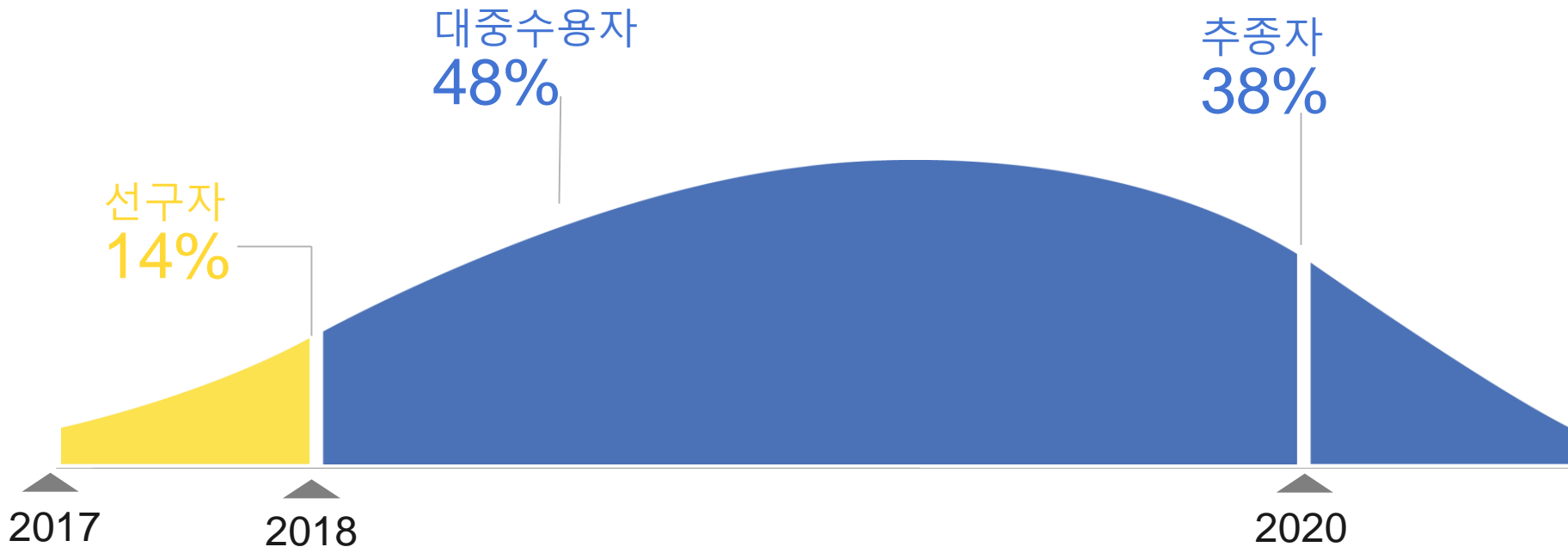
2017년에 대규모로 상용솔루션 구축을 예상하는 선구자 그룹이 블록체인 적용을 선도할 것으로 예상됨



Source: IBM Institute for Business Value analysis

정부기관의 블록체인 활용 전망

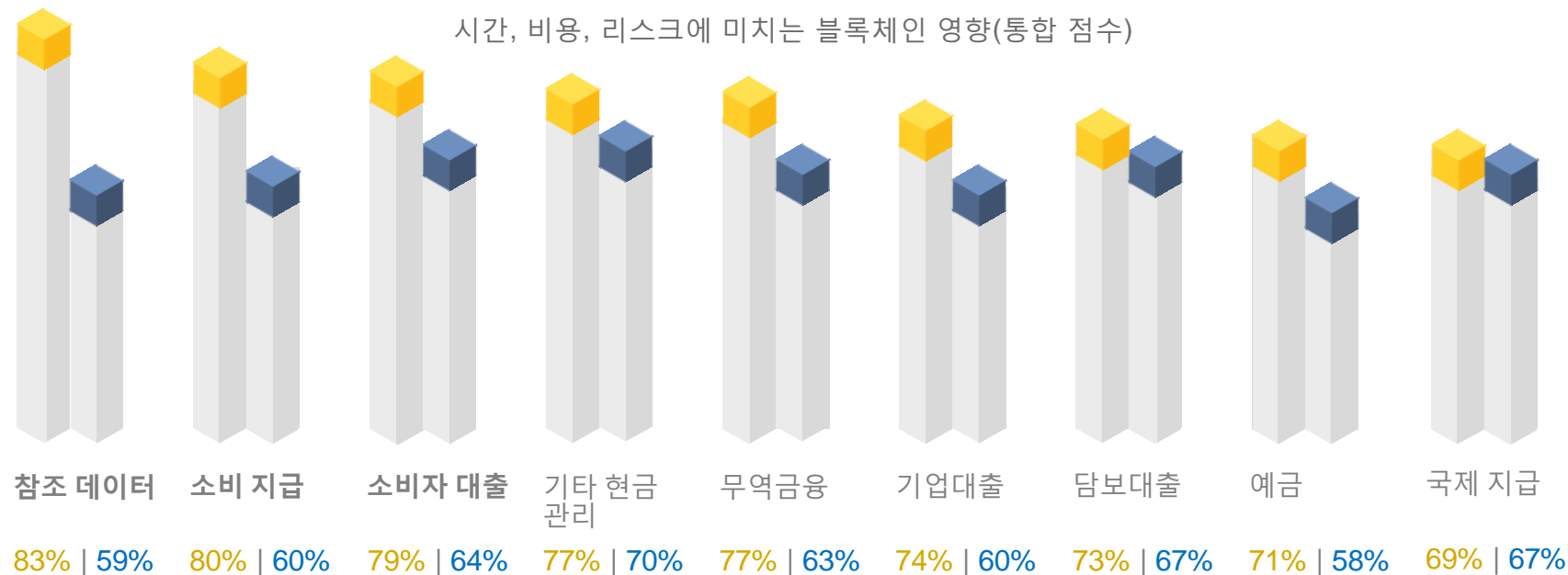
대규모로 상용솔루션 구축을 예상하는 선구자 그룹은 올해 14%, 3년내 62%가
블록체인 적용을 선도할 것으로 예상됨



Source: IBM Institute for Business Value analysis

은행산업의 선구자그룹 비즈니스 모델

선구자 그룹은 참조데이터, 소비 지급, 소비자 대출의 3가지 사업분야가 가장 큰 혜택을 받을 것으로 조사됨

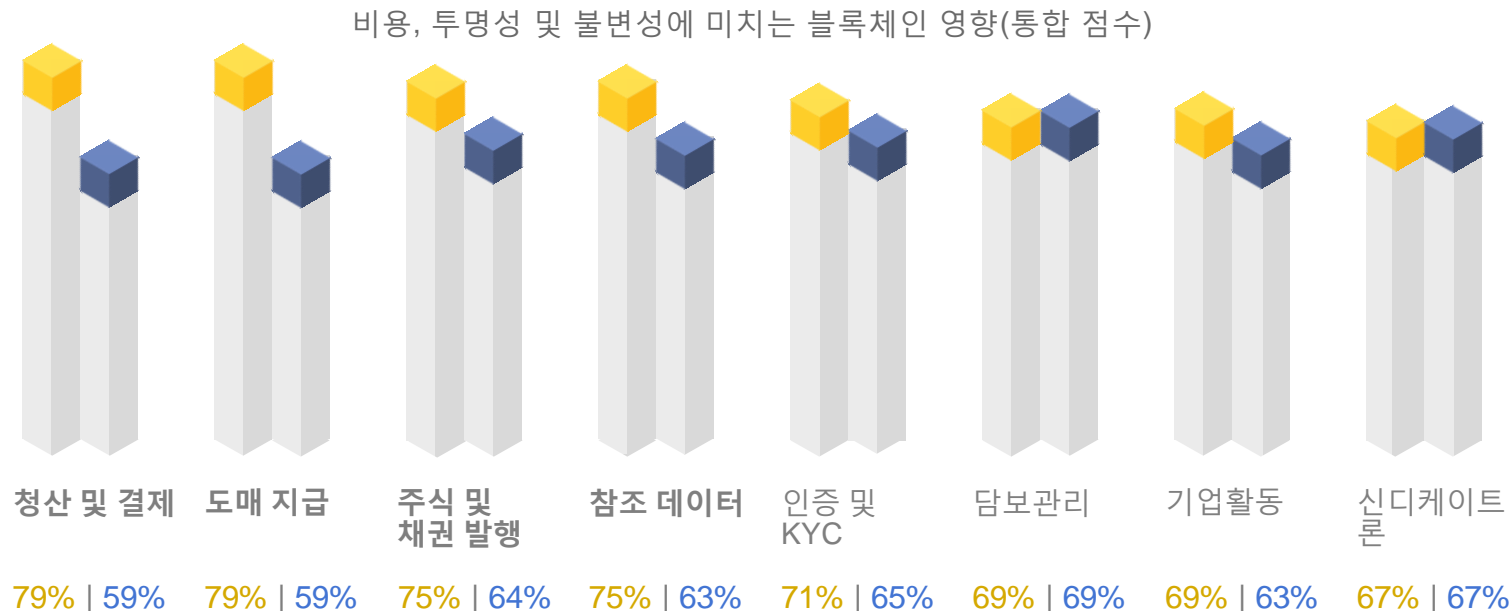


Source: IBM Institute for Business Value analysis,

선구자그룹 기타은행

자본시장의 선구자그룹 비즈니스 모델

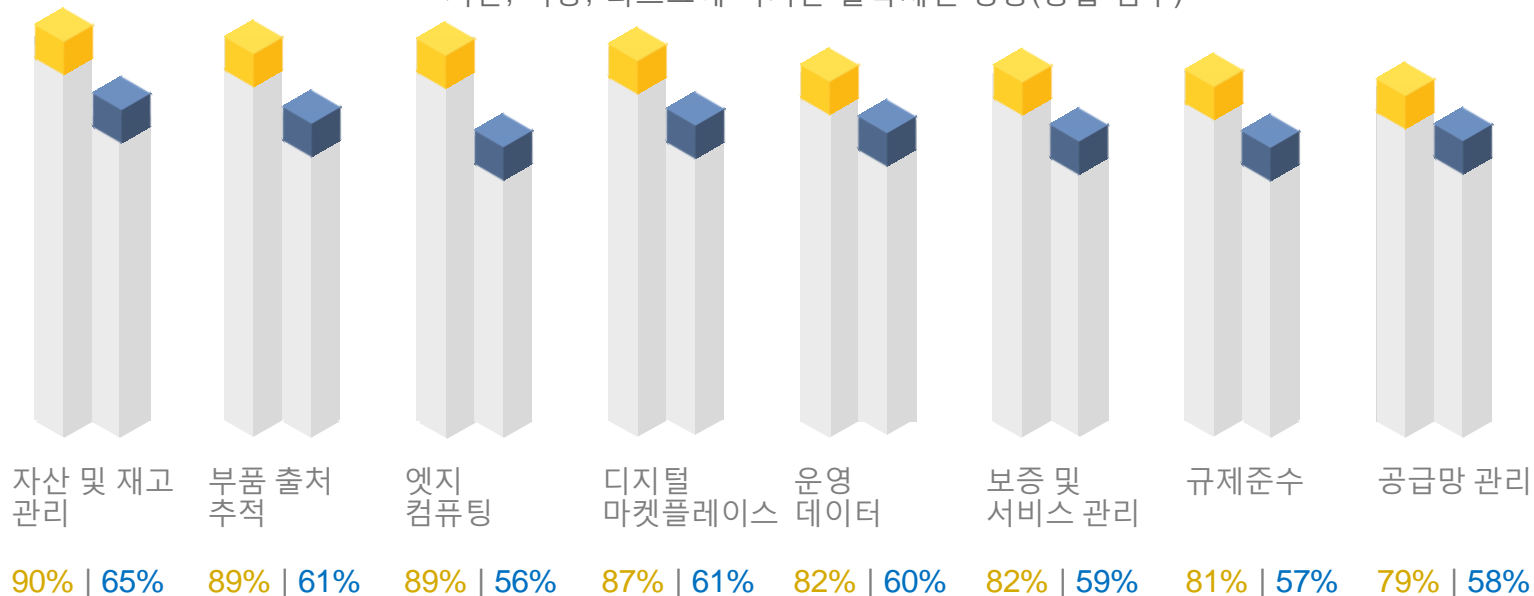
선구자 그룹은 청산 및 결제, 도매 지급(wholesale payment), 주식 및 채권 발행의 3가지 사업분야가 가장 큰 혜택을 받을 것으로 조사됨



전자산업의 선구자그룹 비즈니스 모델

선구자 그룹은 자산 및 재고관리, 부품 출처추적, 엣지 컴퓨팅의 3가지 사업분야가 가장 큰 혜택을 받을 것으로 조사됨

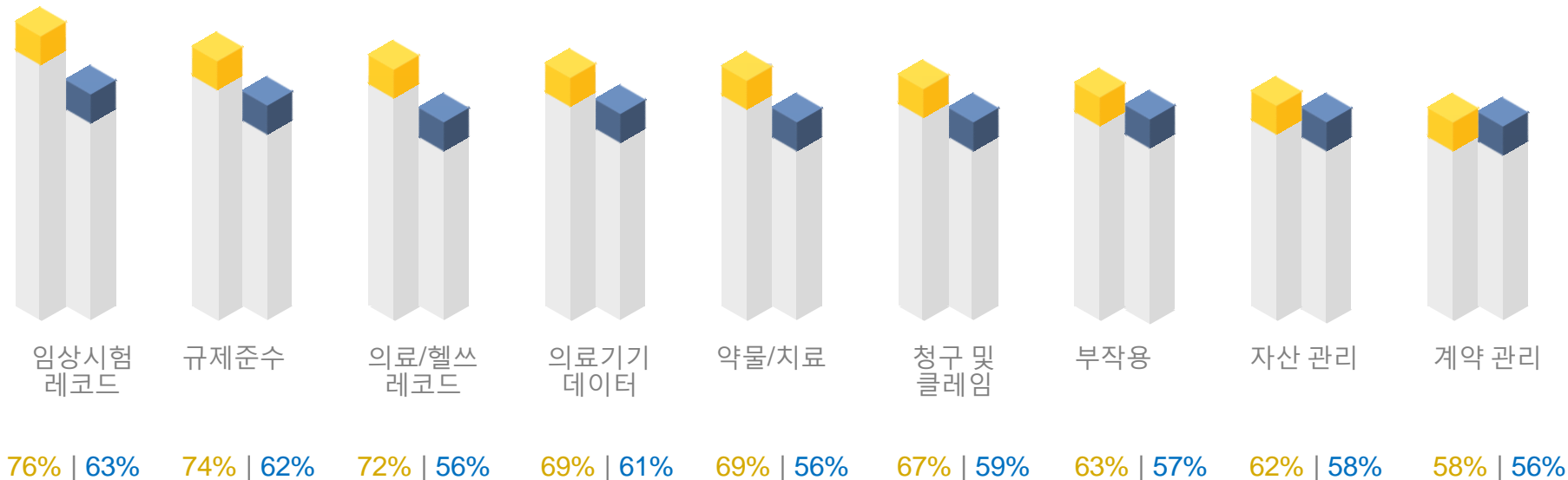
시간, 비용, 리스크에 미치는 블록체인 영향(통합 점수)



헬스산업의 선구자그룹 비즈니스 모델

선구자 그룹은 임상시험 레코드, 규제준수, 의료/헬스 데이터의 3가지 사업분야가 가장 큰 혜택을 받을 것으로 조사됨

시간, 비용, 리스크에 미치는 블록체인 영향(통합 점수)



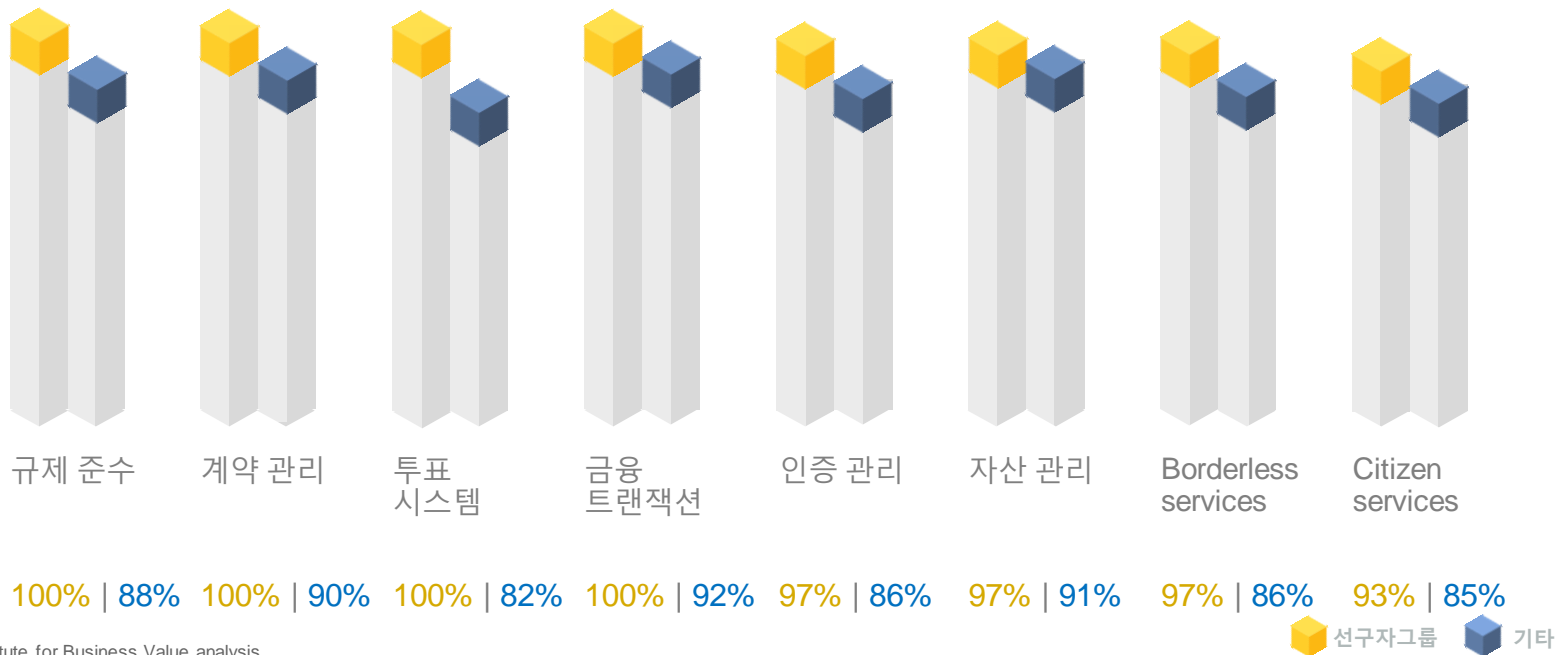
Source: IBM Institute for Business Value analysis,

선구자그룹 기타

정부기관의 선구자그룹 비즈니스 모델

선구자 그룹은 규제 준수, 계약 관리, 투표시스템의 3가지 사업분야가 가장 큰 혜택을 받을 것으로 조사됨

시간, 비용, 리스크에 미치는 블록체인 영향(통합 점수)



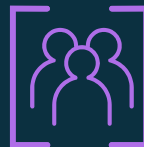
Contents



What is Blockchain?



Why is it relevant for our business?



How can IBM help us apply Blockchain?

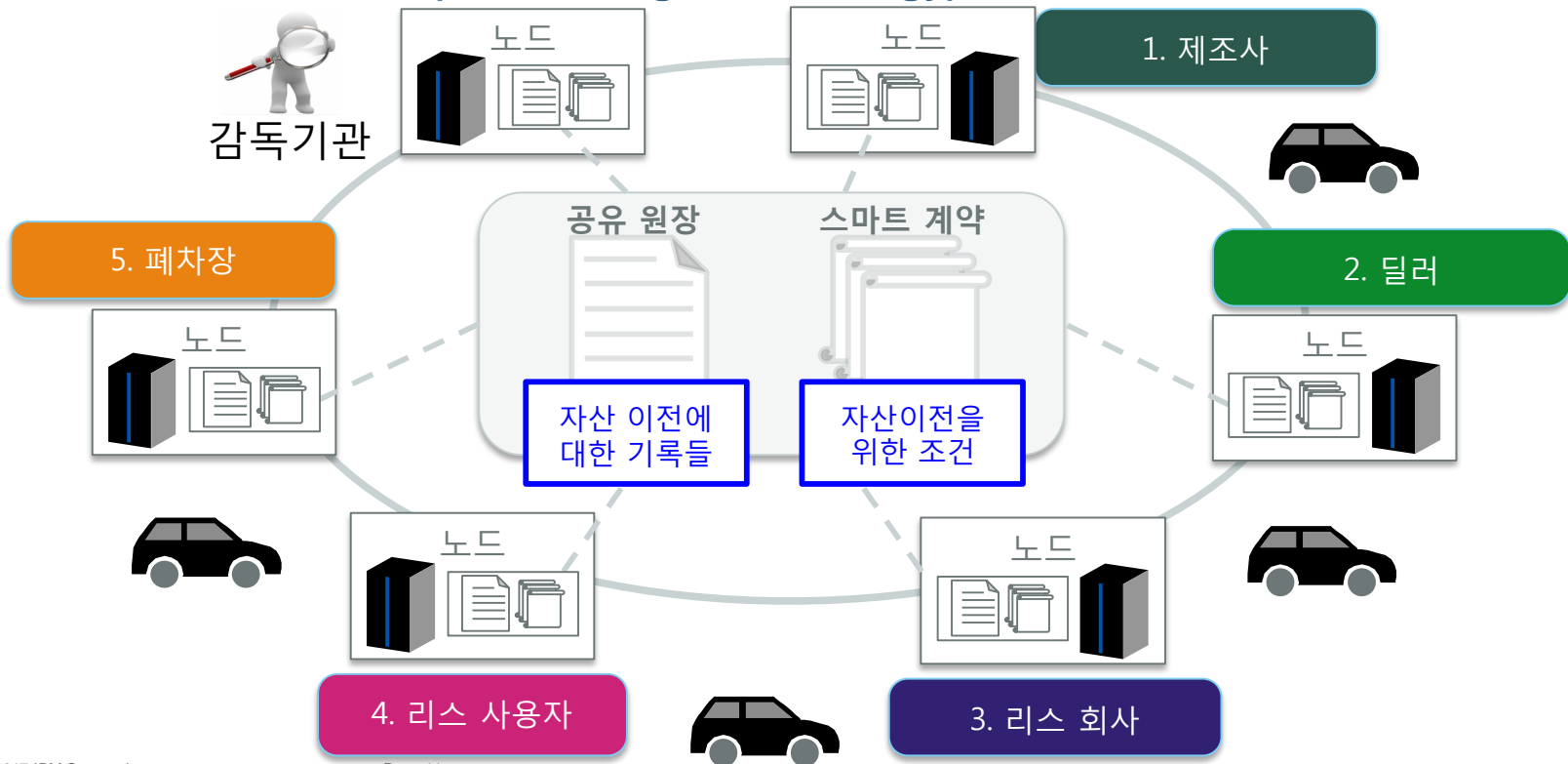


First Projects
Use cases

블록체인이란 ...



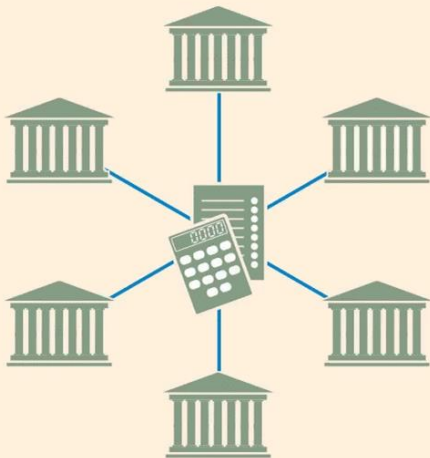
비즈니스 네트워크(Business Network)의 모든 참여자들이 원장(Ledger)을 볼 수 있도록 해주는 **공유 원장 기술 (shared ledger technology)**



현재 시스템

Model 1

Current system



All banks check with central electronic ledger

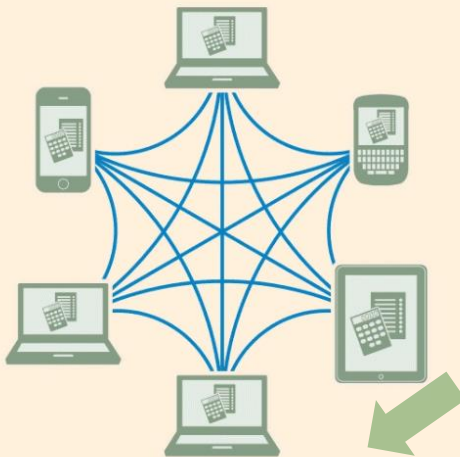
FT

블록체인 종류

Model 2

e.g. Bitcoin

Public blockchain (permissionless)



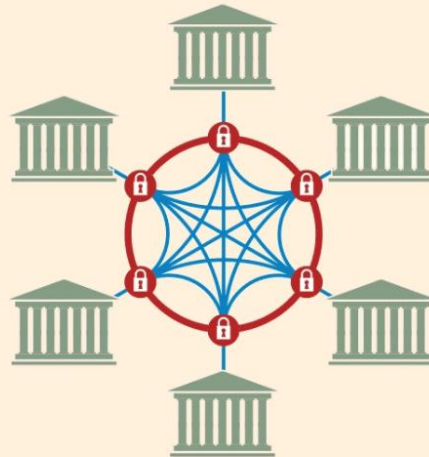
An open network that anybody can access, like the bitcoin model. The digital ledger of transactions is shared, transparent and run by all participants

FT

Model 3

..for business

Private blockchain (permissioned)



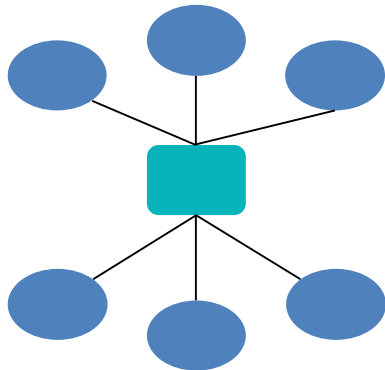
The preferred option of most banks. It is a closed system checking all details and controlling access via invitation

FT

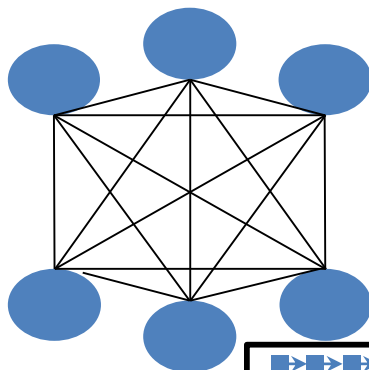
블록체인 요약



Central Network

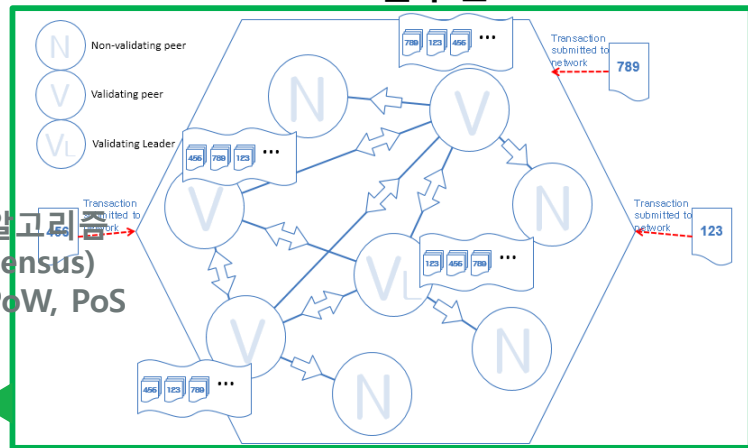


Distribute Network



PBFT 솔루션

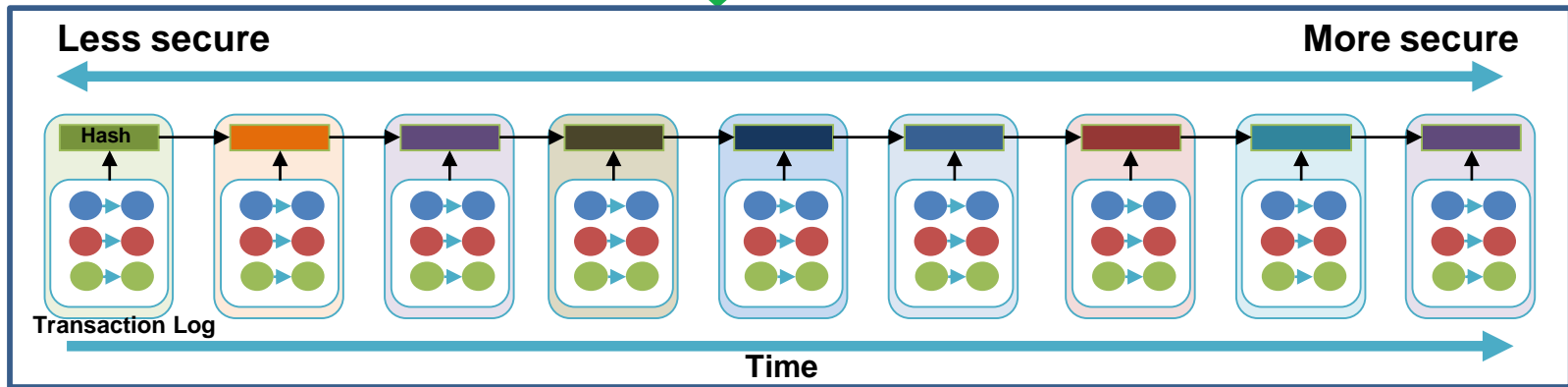
합의 알고리즘
(consensus)
PBFT, PoW, PoS



Less secure

More secure

시간순서상으로
기록된 암호화된
원장



비즈니스를 위한 블록체인...



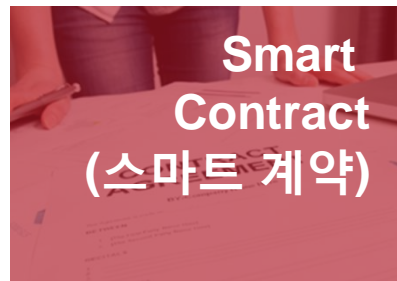
What

비즈니스
네트워크내에 모든
거래가 기록되고
공유됨

Shared
ledger
(공유원장)



Smart
Contract
(스마트 계약)



비즈니스 규칙 및
로직은 계약에
함축되어 트랜잭션
수행시 실행됨

원장은 공유되지만,
참여자의 개인정보는
암호화 기술을 통해서
보호되어야 함

Privacy
(프라이버시)



Consensus
(합의)



검증된 트랜잭션에
대한 네트워크에
참여한 참여자의
동의가 필요함

... 참여확대, 비용절감, 효율성 증대

Contents



What is Blockchain?



Why is it relevant
for our business?



How can IBM help
us apply Blockchain?



First Projects
Use cases



시간 절약

거래처리 시간이
일(days)단위에서 준
실시간 처리



비용 절감

중개자의 오버헤드
및 비용 감소



위험 감소

조작, 사기 및 사이버
범죄 감소



신뢰 확산

공유 프로세스 및
기록을 통해 신뢰
확보

비즈니스를 위한 블록체인 특징

[?] Why

구성요소

특징

Shared
ledger
(공유원장)

Smart
Contract
(스마트 계약)

Privacy
(프라이버시)

Consensus
(합의)

합의 (Consensus)

자산추적성 (Provenance)

불변성 (Immutability)

최종성 (Finality)

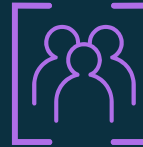
Contents



What is Blockchain?



Why is it relevant
for our business?



How can IBM help
us apply Blockchain?



First Projects
Use cases

리눅스 재단의 하이퍼레저 프로젝트

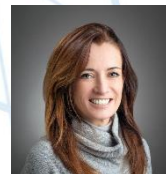


- 리눅스 재단의 Hyperledger 프로젝트는 2015년 12월 17일에 17개 회원사로 시작되었으며, 현재 130개 회원사가 참여하고 있음
- Hyperledger 프로젝트는 전세계적으로 비즈니스 거래가 수행되는 방식을 변혁할 수 있는 분산 원장에 대한 산업 표준에 중요한 기능들을 확인하고 적용하여 블록체인을 발전시키기 위한 협력 프로젝트임
- 오픈 소스, 오픈 표준, 오픈 거버넌스 기반
- 1개의 Active 프레임워크("Fabric")과 4개의 인큐베이터

Enable adoption of shared ledger technology at
a pace and depth not achievable by any one
company or industry



Brian Behlendorf
Executive Director



Blythe Masters
Board Chair



Chris Ferris
TSC Chair

www.hyperledger.org

하이퍼레저 프로젝트 > 회원사



프리미어 회원(18)



일반회원



준회원



Source: <https://www.hyperledger.org/about/members>
Updated April 2017

분산공유원장

비즈니스
네트워크내에 모든
거래가 기록되고
공유됨

Shared
ledger
(공유원장)

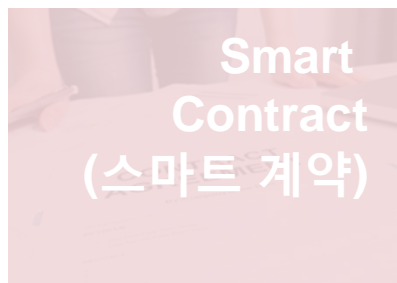


원장은 공유되지만,
참여자의 개인정보는
암호화 기술을 통해서
보호되어야 함

Privacy
(프라이버시)

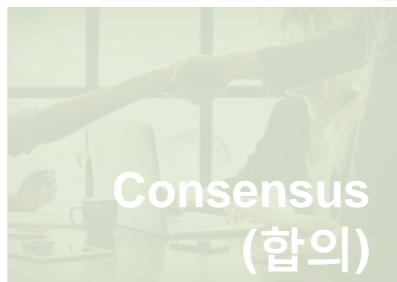


Smart
Contract
(스마트 계약)



비즈니스 규칙 및
로직은 계약에
함축되어 트랜잭션
수행시 실행됨

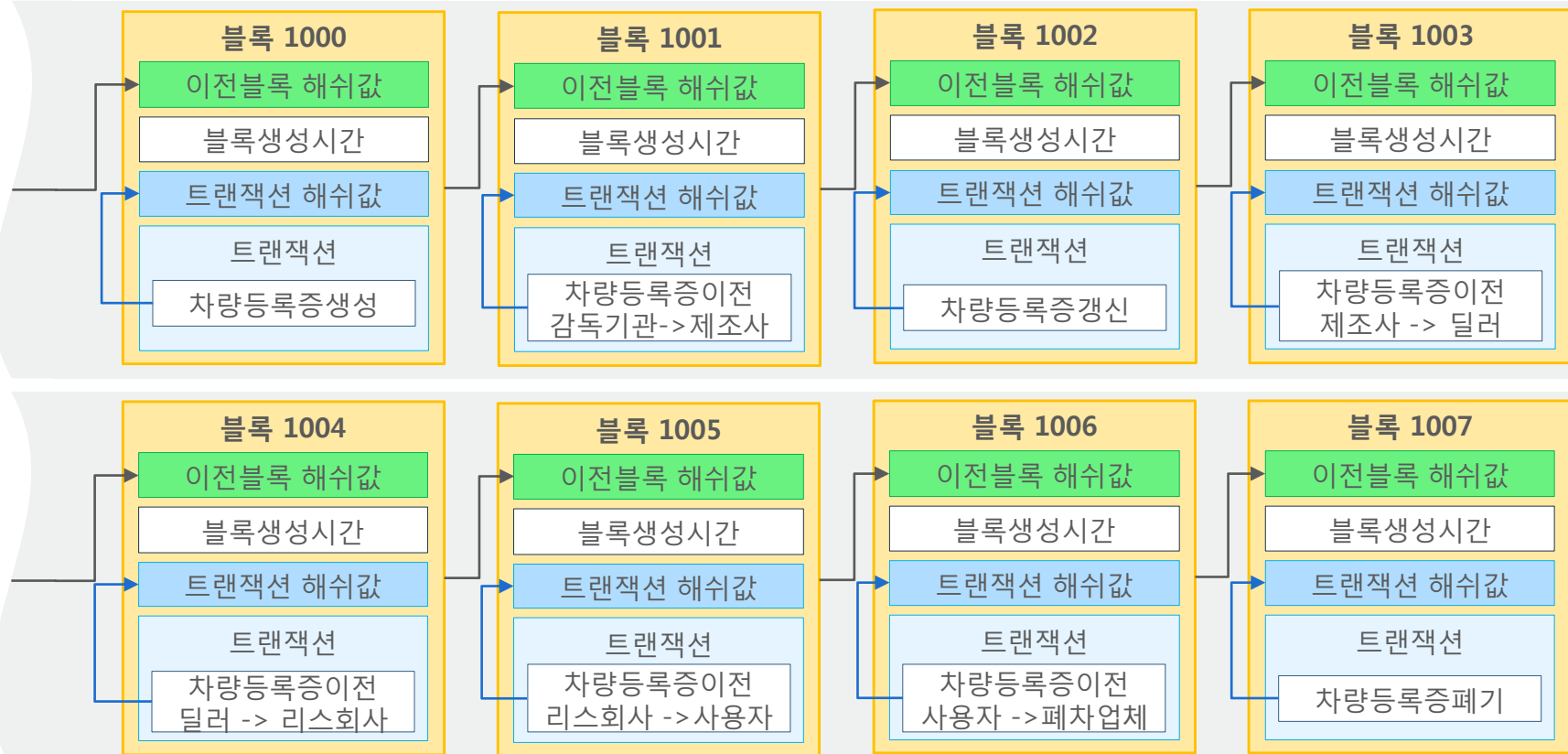
Consensus
(합의)



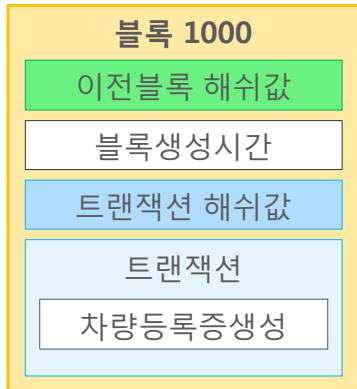
검증된 트랜잭션에
대한 네트워크에
참여한 참여자의
동의가 필요함

... 참여확대, 비용절감, 효율성 증대

분산원장의 구조

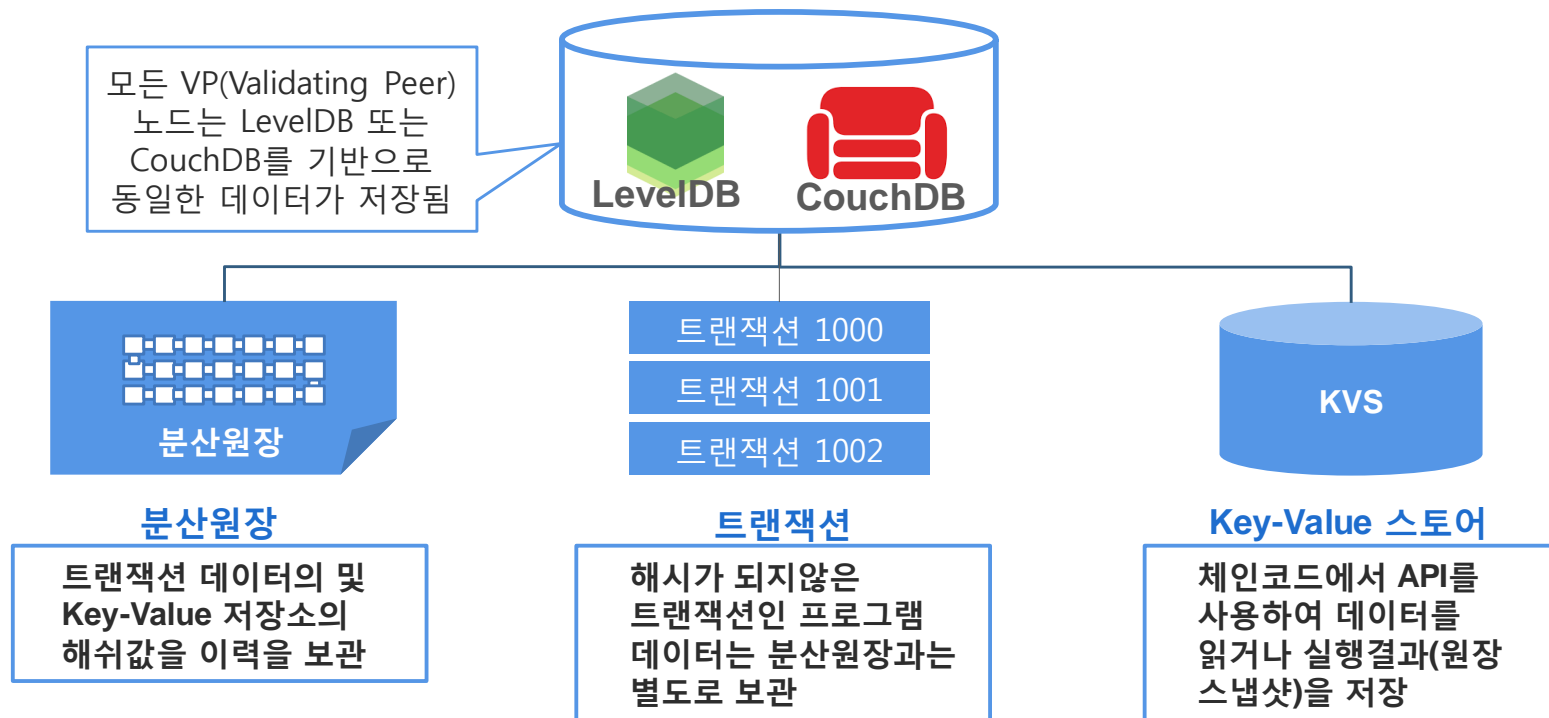


블록 및 트랜잭션



원장 스토리지

검증노드(Validating Peer) 노드는 NoSQL 데이터베이스인 LevelDB나 CouchDB로 key-value 형식의 데이터베이스가 설치되어 있으며, “분산 원장”, “트랜잭션”, “체인 코드(옵션)” 데이터가 저장됨



스마트 계약

비즈니스
네트워크내에 모든
거래가 기록되고
공유됨

Shared
ledger
(공유원장)

원장은 공유되지만,
참여자의 개인정보는
암호화 기술을 통해서
보호되어야 함

Privacy
(프라이버시)

Smart
Contract
(스마트 계약)

비즈니스 규칙 및
로직은 계약에
함축되어 트랜잭션
수행시 실행됨

Consensus
(합의)

검증된 트랜잭션에
대한 네트워크에
참여한 참여자의
동의가 필요함

... 참여확대, 비용절감, 효율성 증대

스마트계약(Smart Contract)/체인코드(Chaincode)

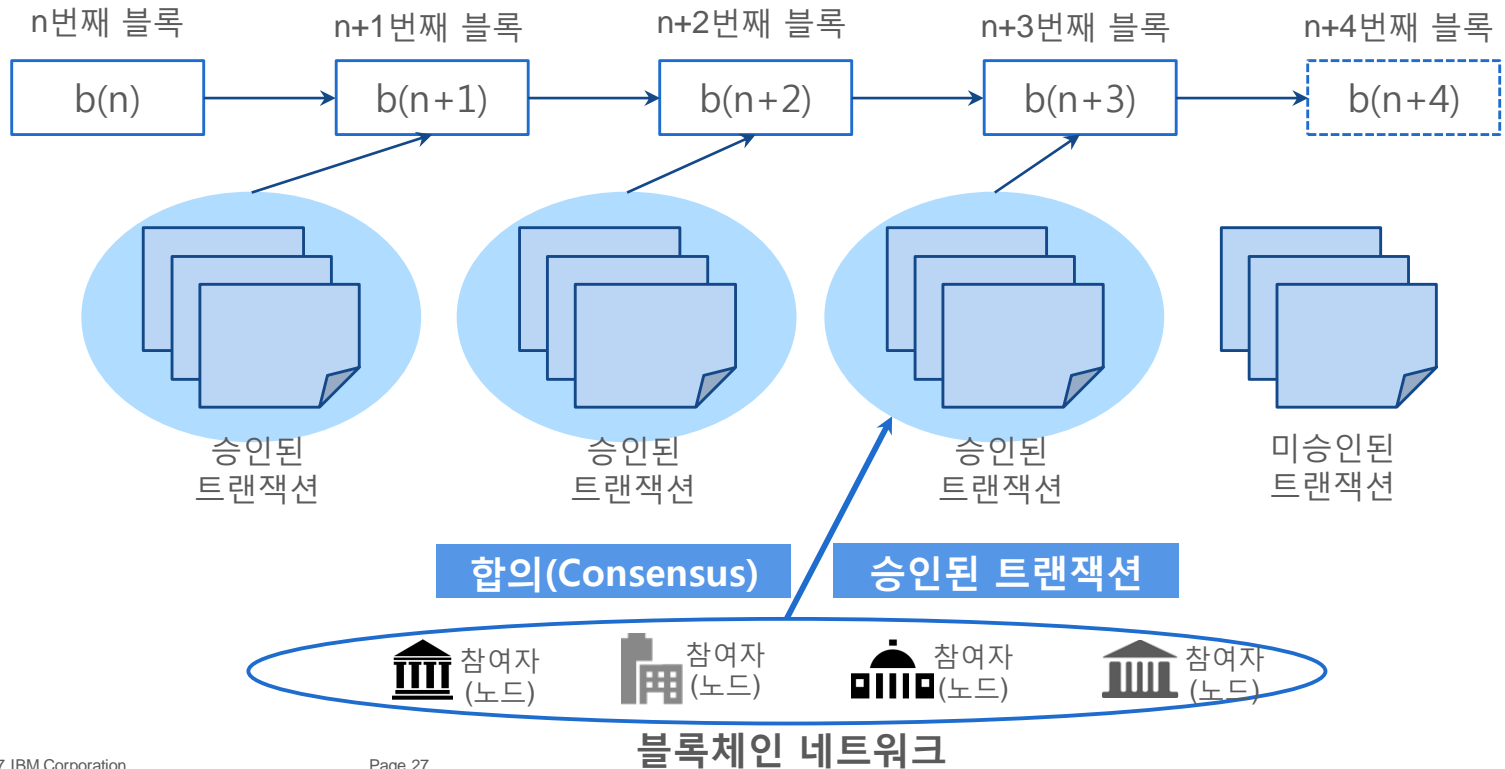
현대화된 암호화 원장(Ethereum, Hyperledger)들은
스마트계약이나 체인코드를 지원하는 것으로 목표로 하고 있음

A smart contract is an **event driven program**, with state, which runs on a replicated, shared ledger and which can take custody over assets on that ledger. [Swanson2015]

“Smart contract” → (replicated) state machine

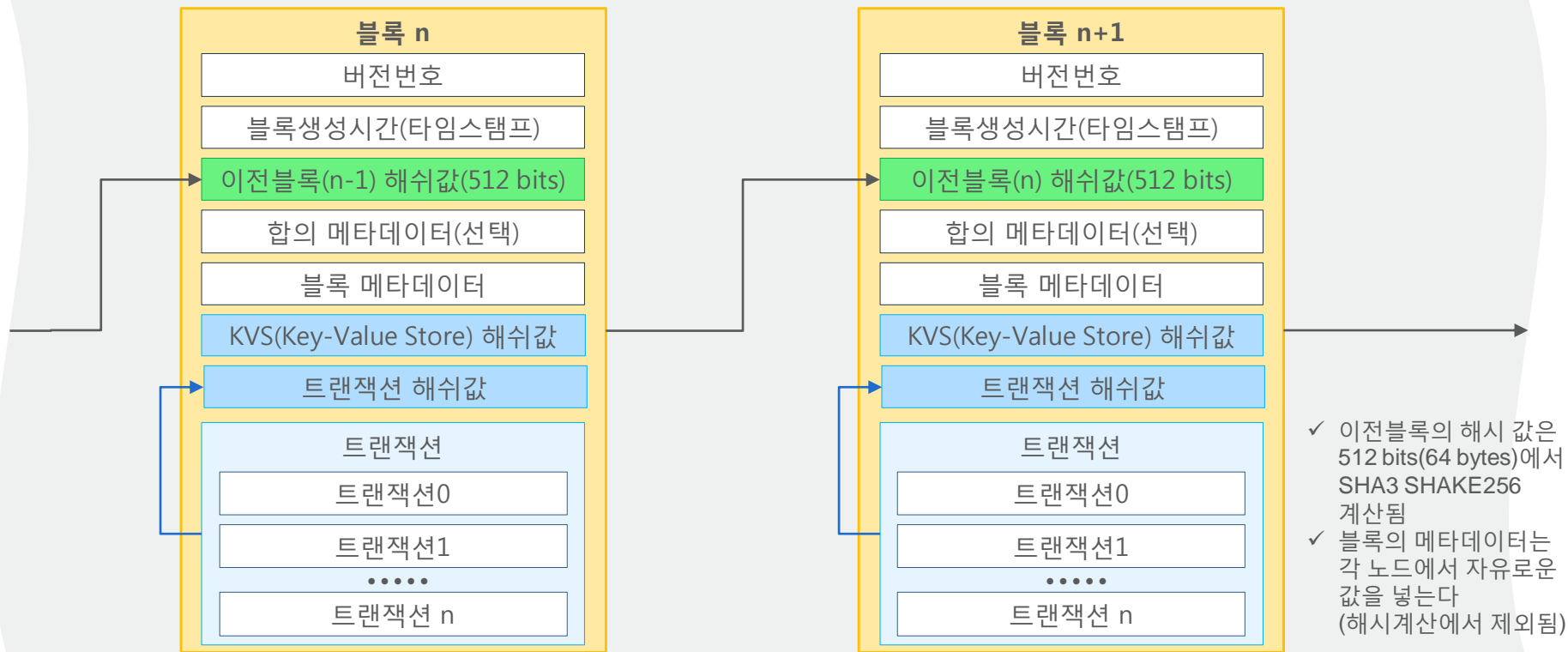
블록체인 참여자 합의에 의해 업데이트됨

합의(Consensus)라는 시스템 기반의 합의 형성 구조는 블록체인을 적절히 사용하여 운용하는 데 매우 중요한 개념임

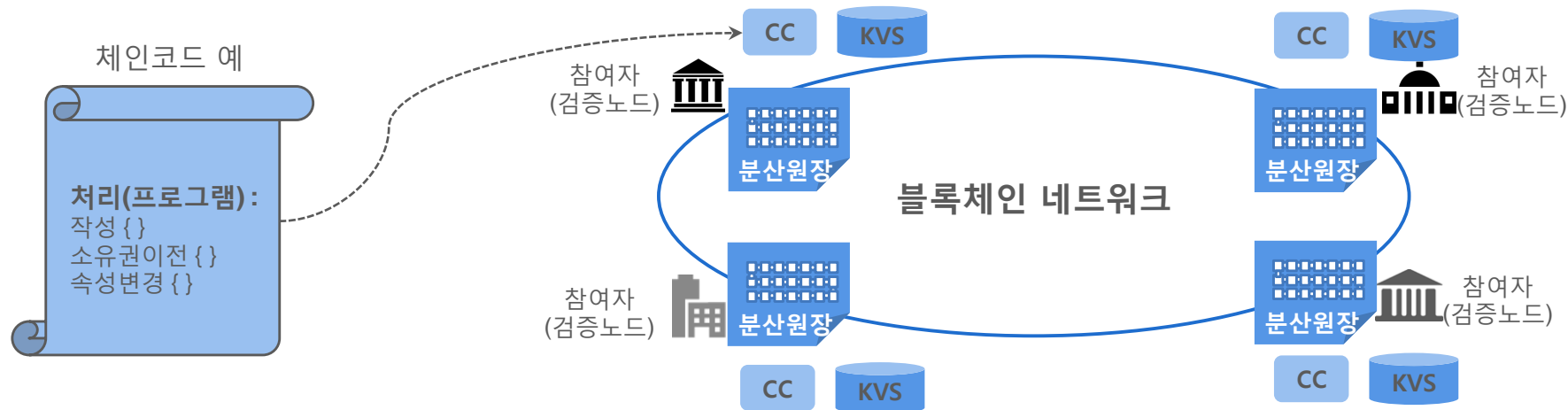


하이퍼레저 패브릭의 블록 구조

암호화 해쉬함수를 사용하여 블록구조를 저장함으로써 과거의 트랜잭션 정보가 훼손되지 않도록 보호함.
블록구조는 Bitcoin의 경우와 거의 동일하지만, Key-Value Store(KVS)의 해쉬값도 기록되는 점이 특징임



체인코드(스마트계약 실행기반)



- ✓ 체인코드(프로그램)는 모든 검증노드에 배포됨
- ✓ 검증노드는 체인코드를 Docker 컨테이너에 배포하고 실행함
- ✓ 하이퍼레저 패브릭은 『트랜잭션』=『체인코드 (스마트 계약)의 실행』
- ✓ 자산의 최신 상태는 Key-value 저장소에 저장함
- ✓ 계약 조건이나 규칙을 비즈니스 로직으로 스마트 계약에 구현함으로써 항상 합의 된 규칙에 따라 처리가 수행됨

검증노드의 구성



참여기관
(검증노드의 구성)



분산원장

블록체인

CC

CC

.....

CC

체인코드
(스마트 계약)

KVS

Key-Value Store
("chaincode state",
"world state"
이라고도 명명됨)

분산원장

- ✓ 블록체인 자체
- ✓ 트랜잭션 (스마트 계약의 처리 호출)가 로그처럼 기록됨

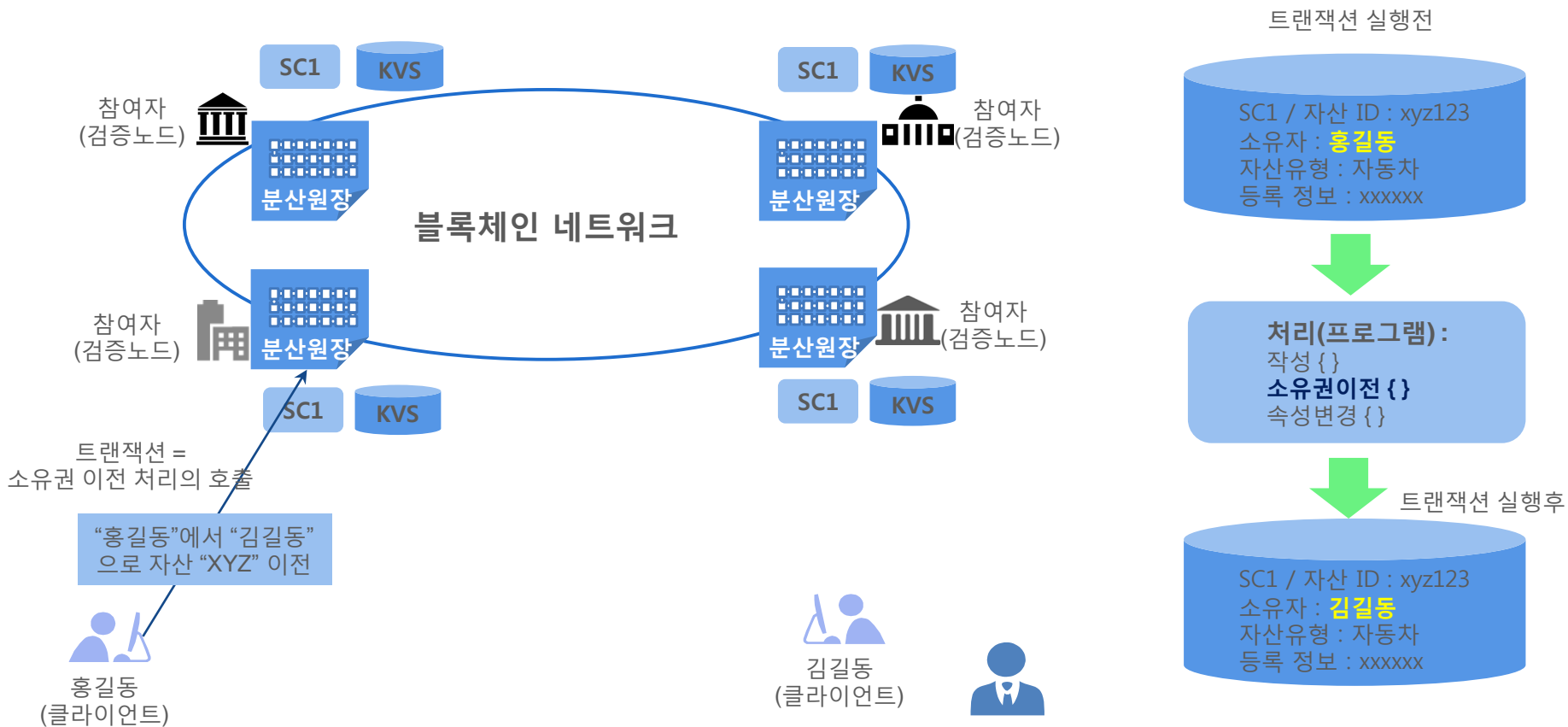
체인코드(스마트계약)

- ✓ 트랜잭션을 계기로 실행되는 프로그램
- ✓ 똑같은 체인코드가 모든 검증노드에 배포되어 샌드박스(Docker 컨테이너)에서 실행됨
- ✓ 사용 목적별로 여러 체인코드가 만들어짐

Key-Value Store

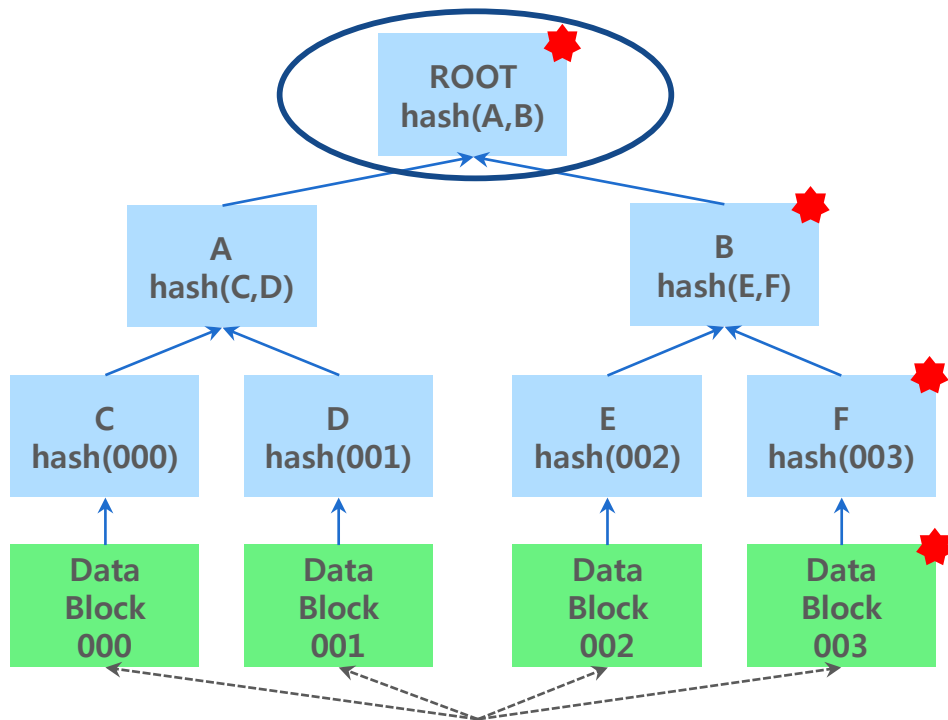
- ✓ 트랜잭션을 실행 한 결과 얻어지는 "최신 상태"를 기록함
 - Key = chaincode ID + cKey
 - Value = 임의의 데이터
- ✓ 모든 검증노드에서 동일한 내용을 갖고, 그 해쉬값이 블록 체인에 기록됨
- ✓ 변경 상태(자산의 이전등)가 관리됨

트랜잭션에 의해 스마트계약에 정의된 작업이 수행됨



Key-Value Store의 해쉬 계산방법

블록체인에 기록되는 값



개별 Key-Value 데이터 쌍

트리구조에서 해쉬값을 계산하고 있기 때문에 KVS의 데이터가 변경된 경우에도 최소한의 해쉬계산에 전체의 해쉬값을 계산함

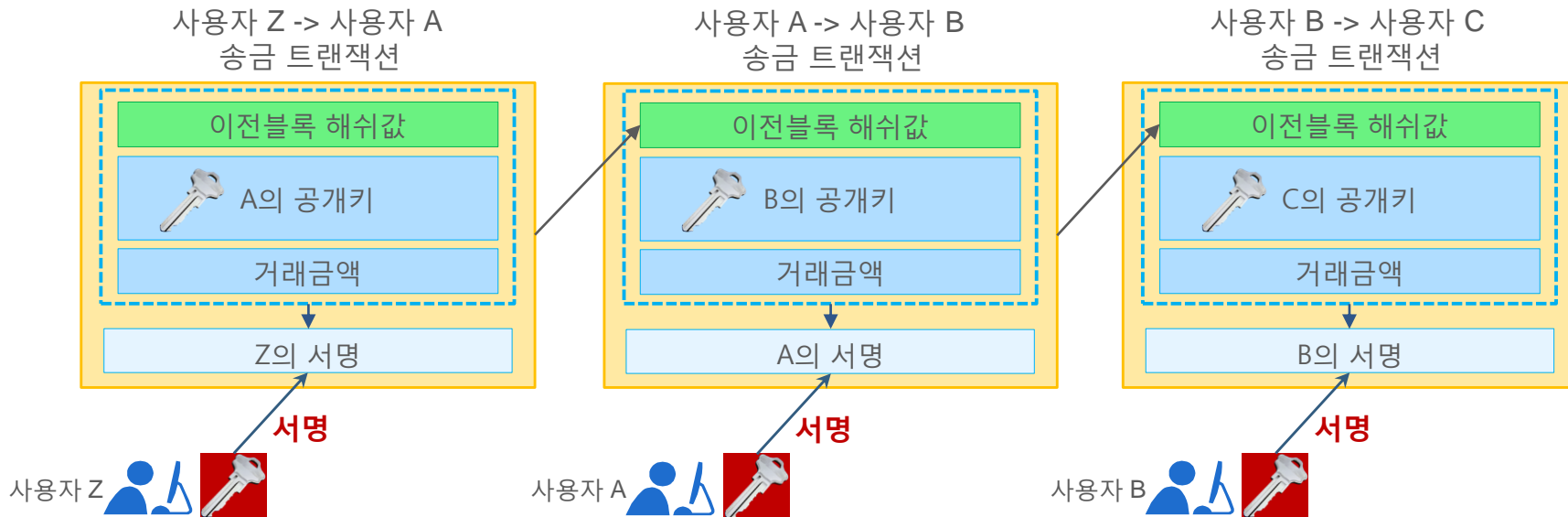
- ✓ 머클 해쉬 트리(Merkle Hash Tree) 함수 사용

★ “Data Block 003”이 변경된 경우에 해쉬값을 다시 계산 이루어지는 점

비트코인(Bitcoin) 트랜잭션 비교(1/3)

비트코인(Bitcoin)을 가지고 A가 B에게 송금하는 경우

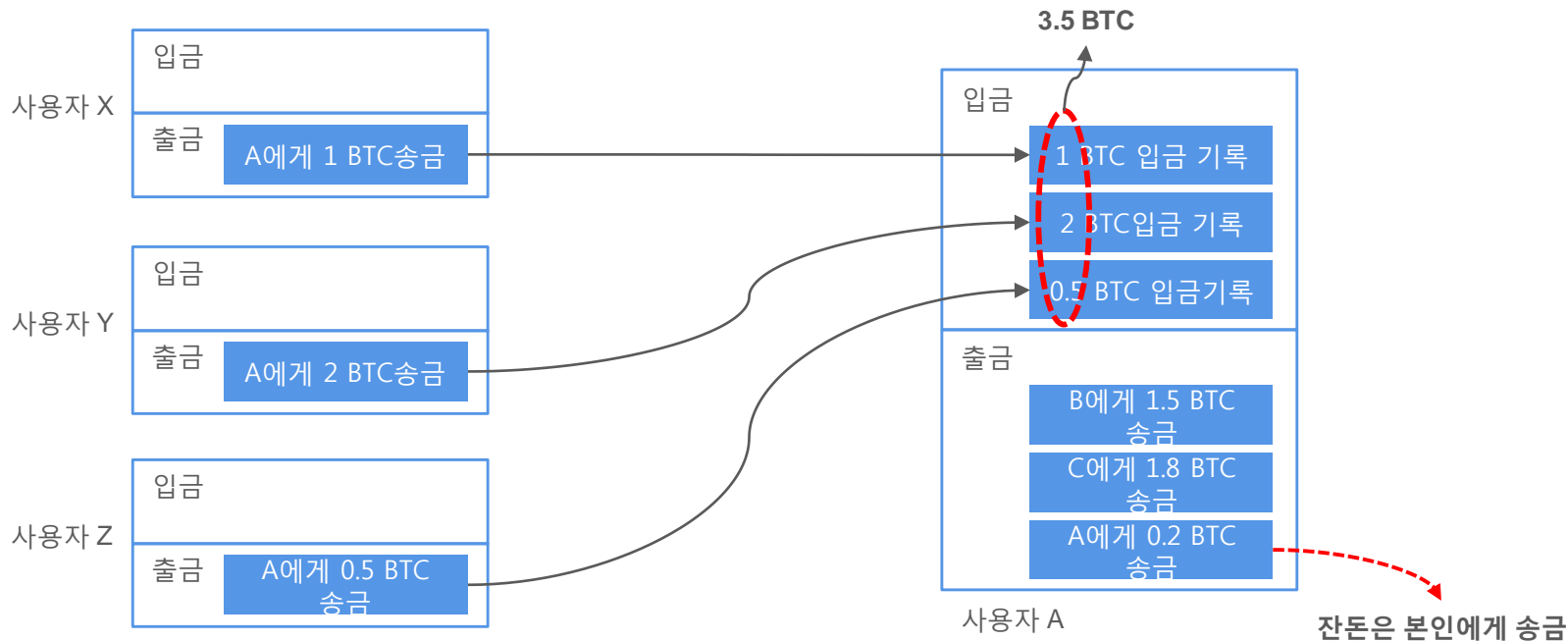
- ✓ A는 "B에게 몇 비트 코인을 보냅니다"라는 트랜잭션에 서명을 함으로써 성립
- ✓ 이때 A는 자신이 가지고있는 비트코인을 받은 트랜잭션에 대한 참조와 수령자인 B의 공개키를 붙인 후에 자신의 개인키로 서명



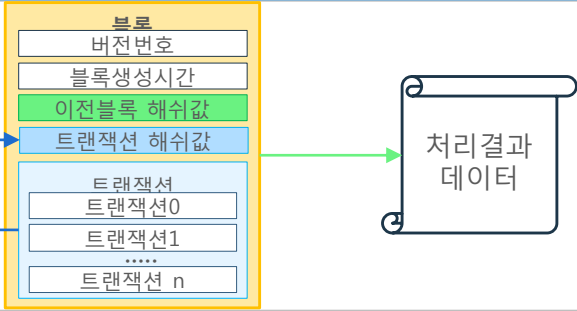
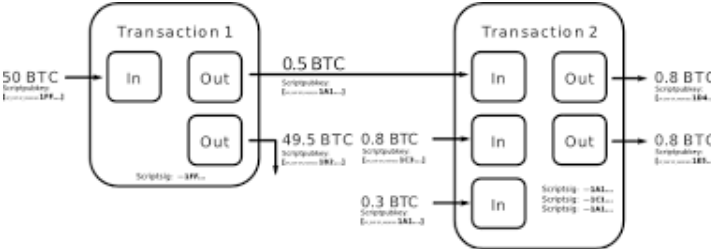
비트코인(Bitcoin) 트랜잭션 비교(2/3)

비트코인(Bitcoin)은 Unspent Transaction Output (UTXO) 방식을 사용

- ✓ 하나의 주소 (지갑)에 들어있는 여러 트랜잭션의 입금울 모아 여러 사람 주소로 전송하는 것이 가능



비트코인(Bitcoin) 트랜잭션 비교(3/3)

분류	Smart Contract형	UTXO형
구조		
요약	처리에 대한 요청 메시지를 트랜잭션으로 표현하고 처리 결과를 유지하기 위해 현재의 상태 (잔액 및 보유 자산 등)를 확인하기 위해 과거의 트랜잭션을 추적 할 필요가 없음	트랜잭션 데이터의 입력, 출력 형태로 표현하고, 현재 상태 (잔액 및 보유 자산 등)를 확인하기 위해서는 과거의 트랜잭션을 추적 할 필요가 있음
용도	Smart Contract로 트랜잭션 처리를 기술 및 실행함으로써, 다양한 종류의 트랜잭션을 처리 할 수 있음	디지털 화폐의 이전 등 특정 유형의 트랜잭션에 제한됨
잘못 사용된 트랜잭션 검증	Smart Contract에 원칙적으로 잘못 사용된 트랜잭션에 대한 확인을 기술하고 Smart Contract실행시 오류로 처리 함	합의를 수행할 때 트랜잭션 내용의 정당성(예 :이중 사용)에 대해서도 각 노드가 검증
대표적인 구현 예	하이퍼레저(Hyperledger), 이더리움(Ethereum)	비트코인(Bitcoin), DAH, chain.com

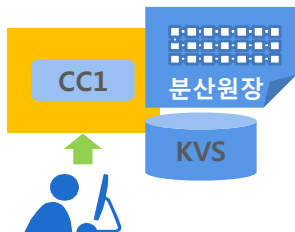
체인코드 관련처리

◆ 주요 체인코드 관련 처리 (가상머신에서 수행)



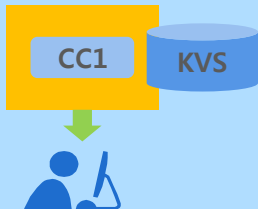
배포(Deploy)

- ✓ 소스코드에 따라 체인코드를 등록
- ✓ 원장에 기록됨(새로 배치했다는 정보가 기록됨)



호출(Invoke)

- ✓ 체인코드를 실행함
- ✓ KVS에 데이터를 읽고 저장함
- ✓ 원장에 기록됨

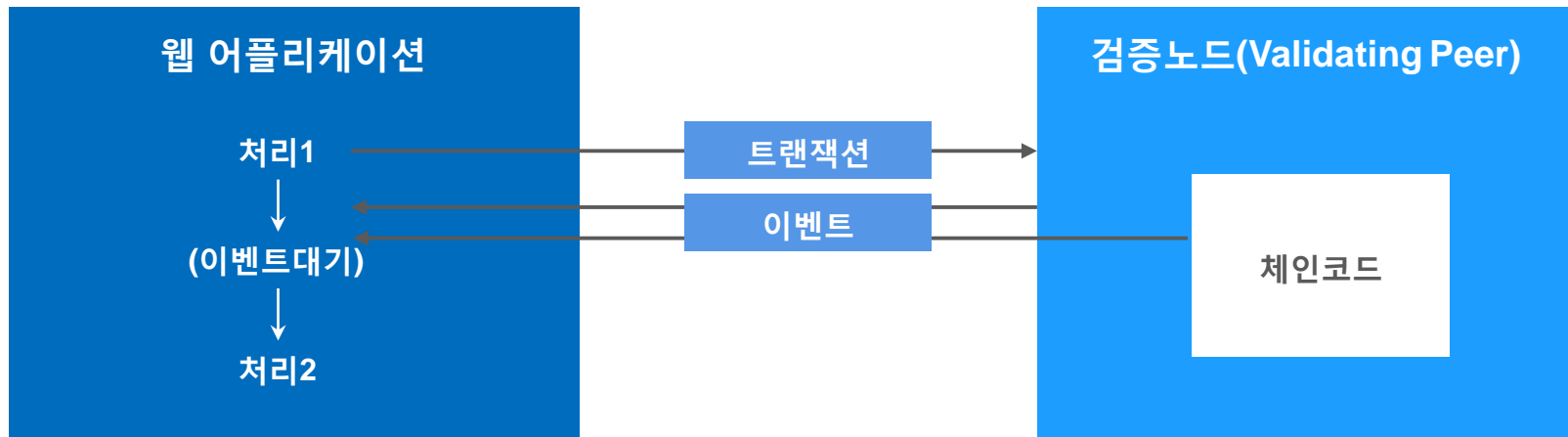


조회(Query)

- ✓ 체인코드에 데이터 조회함
- ✓ KVS에서 데이터 읽기만 수행함
- ✓ 원장에 기록하지 않음

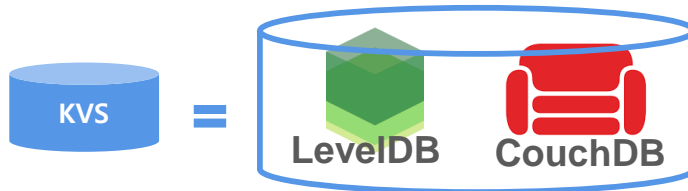
이벤트 허브

검증노드(Validating Peer)와 체인코드는 응용 프로그램이 완료를 기다렸다가 동작을 수행하기 위해 네트워크에 이벤트를 실행할 수 있음. 미리 정의 된 이벤트도 있지만, 체인 코드 자체의 이벤트를 정의하고 발행 할 수 있음



체인코드의 상태를 저장하는 데이터베이스

Hyperledger의 체인코드는 데이터베이스로 키(Key)와 값(Value)의 쌍으로 데이터를 등록하는 KVS 방식의 LevelDB 또는 CouchDB를 채용하고 있으며, RocksDB은 오픈소스 데이터베이스로 ORACLE, MySQL 과 같은 관계형 데이터베이스와 비교하여 데이터베이스 스키마를 유연하게 변경할 수 있는것이 특징임.



전체 검증노드(Validating Peer)에 존재하고 동일한 데이터가 저장됨



가상머신에서
체인코드
실행

PutState ("Asset-1-Holder",
"Manufacturer")

GetState ("Asset-2-Holder")

"Lease"

키(Key)	값(Value)
CC1:Asset-1-Manufacturer	Hyundai
CC1:Asset-1-Holder	Manufacturer
CC1:Asset-1-Price	60,000,000 Won
CC1:Asset-2-Manufacturer	Toyota
CC1:Asset-2-Holder	Lease
CC1:Asset-2-Price	45,000,000 Won
.....

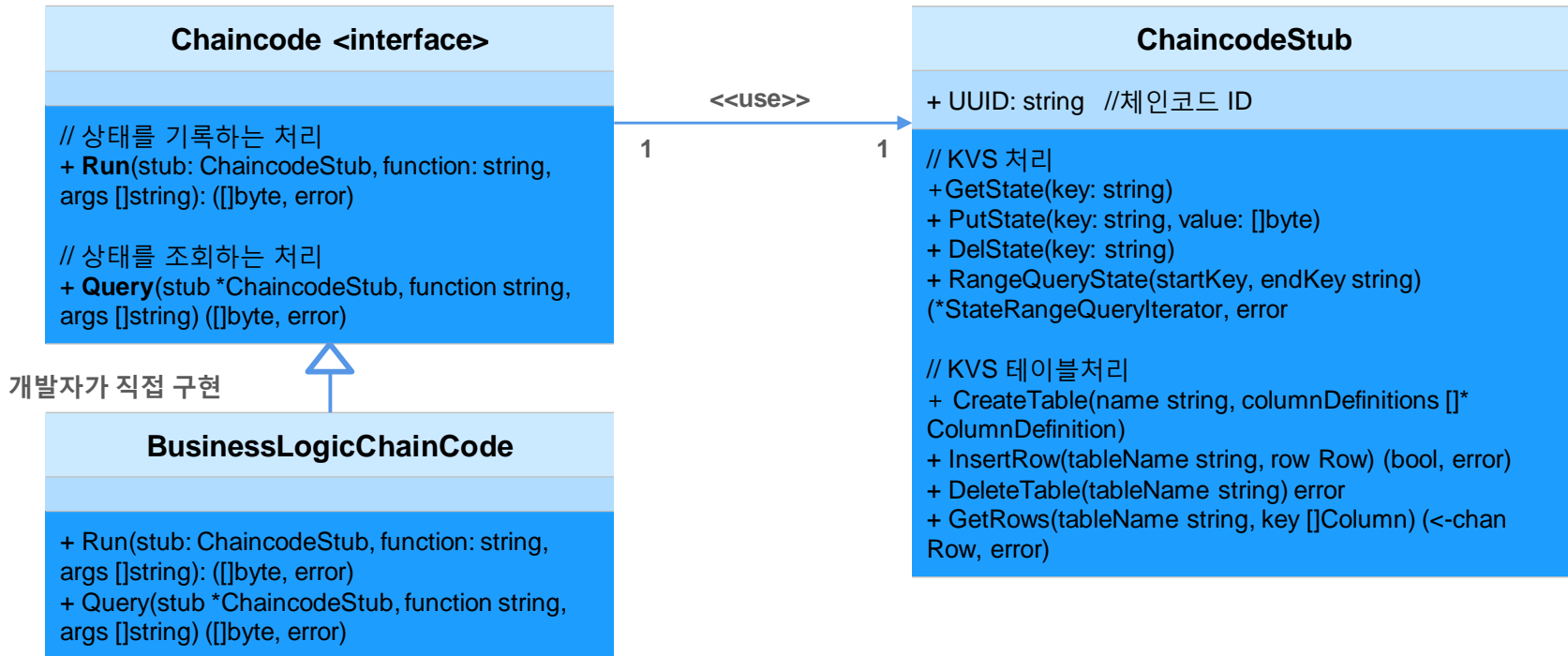
키의 체인코드의 ID가 접두어로 지정되기
때문에 다른 체인코드의 데이터끼리
충돌하는 것은 아님

체인코드 구현

- ✓ 체인코드 개발자는 Chaincode 인터페이스를 구현하는 형태로 비즈니스 로직을 구현함
- ✓ 체인코드는 ChaincodeStub 클래스의 인스턴스를 호출하는 형태로 데이터베이스에 접근함

체인코드 사양 설명

KVS와 통신 담당



체인코드 구현 예제(단순 송금시스템 : 호출)



Go

```
// Run callback representing the invocation of a chaincode
// This chaincode will manage two accounts A and B and will transfer X units
// from A to B upon invoke
func (t *SimpleChaincode) Run(stub *shim.ChaincodeStub, function string,
    args []string) ([]byte, error) {
```

```
    // Handle different functions
    if function == "init" {
        // Initialize the entities and their asset holdings
        return t.init(stub, args)
    } else if function == "payment" {
        // Transaction makes payment of X units from A to B
        return t.payment(stub, args)
    } else if function == "delete" {
        // Deletes an entity from its state
        return t.delete(stub, args)
    }
}
```

함수이름에 따라
처리 배분

지불을 "payment"
함수로 구현
(다음페이지에서 설명)

```
    return nil, errors.New("Received unknown function invocation")
}
```


구현 예제(payment 함수)



Go

```
// Transaction makes payment of X units from A to B
func (t *SimpleChaincode) invoke(stub *shim.ChaincodeStub, args []string)
([]byte, error) {
```

```
var A, B string    // Entities
var Aval, Bval int // Asset holdings
var X int          // Transaction value
var err error
```

```
A = args[0]
B = args[1]
```

변수 선언과 전처리

```
// Get the state from the ledger
```

```
Avalbytes, err := stub.GetState(A)
Aval, _ = strconv.Atoi(string(Avalbytes))
```

1. KVS에서 사용자 A의 잔액 조회

```
Bvalbytes, err := stub.GetState(B)
Bval, _ = strconv.Atoi(string(Bvalbytes))
```

2. KVS에서 사용자 B의 잔액 조회

```
// Perform the execution
```

```
X, err = strconv.Atoi(args[2])
Aval = Aval - X
Bval = Bval + X
```

3. A에서 B로 송금

```
// Write the state back to the ledger
```

```
stub.PutState(A, []byte(strconv.Itoa(Aval)))
stub.PutState(B, []byte(strconv.Itoa(Bval)))
```

4. 잔액을 KVS에 저장

```
return nil, nil
```

```
}
```

체인코드 구현 예제(단순 송금시스템 : 조회)



Go

```
// Query callback representing the query of a chaincode  
func (t *SimpleChaincode) Query(stub *shim.ChaincodeStub, function string,  
    args []string) ([]byte, error) {
```

```
    var A string // Entities  
    var err error
```

```
    A = args[0]
```

변수 선언과 전처리

```
    // Get the state from the ledger
```

```
    Avalbytes, err := stub.GetState(A)
```

```
    return Avalbytes, nil
```

1. KVS에서 A의 잔액조회

2. 값을 노드에 반환

```
}
```

분산 합의

비즈니스
네트워크내에 모든
거래가 기록되고
공유됨

Shared
ledger
(공유원장)

Smart
Contract
(스마트 계약)

비즈니스 규칙 및
로직은 계약에
함축되어 트랜잭션
수행시 실행됨

원장은 공유되지만,
참여자의 개인정보는
암호화 기술을 통해서
보호되어야 함

Privacy
(프라이버시)

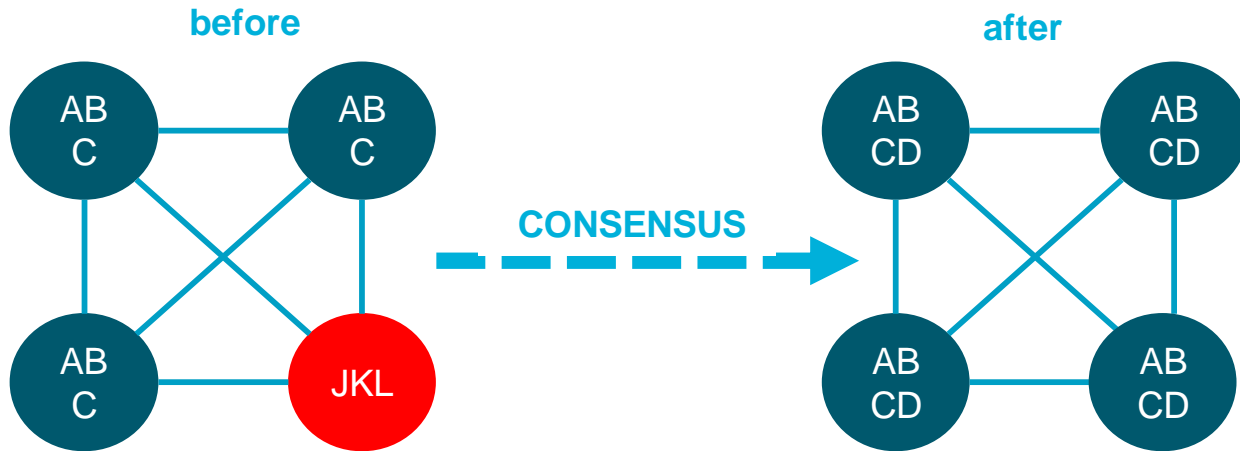
Consensus
(합의)

검증된 트랜잭션에
대한 네트워크에
참여한 참여자의
동의가 필요함

... 참여확대, 비용절감, 효율성 증대

합의(Consensus)란...

합의는 블록체인 네트워크에있는 모든 참여자의 원장(블록 및 상태)이 일관성이 있는지 확인하는 메커니즘



- 거래 및 거래 실행 순서에 대한 동의
- 동일한 원장을 유지하기 위하여 검증 참여자들의 상태를 동기화
- 거래원장이 일치하지 않는 참여자 노드의 상태 수정
- 악의적인 참여자 노드들은 격리

peer/node



합의(Consensus) 관리자

합의(Consensus)라는 시스템 기반의 합의 형성 구조는 블록체인을 적절히 사용하여 운용하는 데 매우 중요한 개념임

- 1) 블록체인은 참가자 합의에 의해 업데이트 됨
- 2) 합의 알고리즘은 여러가지 존재 : 작업증명(PoW), 지분증명(PoS), PBFT
- 3) 분산 환경에서의 멀티 파티 컨센서스 (합의) 형성 알고리즘
- 4) 작업증명 (Proof-of Work) 알고리즘의 문제 (1) : 최종성(Finality)의 부족
- 5) 작업증명 (Proof-of Work) 알고리즘의 문제 (2) : 51 % 공격 문제
- 6) 플러그인 구조

합의 알고리즘 종류

관리주체	관리자가 없음	여러조직	단일조직
네트워크 형태	퍼블릭(Public) 형	컨소시움(Consortium) 형	프라이빗(Private) 형
참여자	자율적	허가적(Permissioned)	
	불특정 사용자가 참여할 수 있기 때문에, 악의를 가진 사용자가 참여할 수 있음	참여자의 신원이 확인되고, 신뢰할 수 있음	
분산합의 알고리즘 (Consensus)	작업증명(Proof of work) 및 지분증명(Proof of stake) 알고리즘 (마이닝)	특정 노드(검증자)에 의한 검증 (분산합의 알고리즘)	
	<ul style="list-style-type: none"> ✓ 전력소비가 많음 ✓ 최종성이 없음 ✓ 51% 공격 문제 	<ul style="list-style-type: none"> ✓ 전력소비를 최소화 ✓ 최종성이 있음 ✓ 경량화 되어 있으며, 빠르게 처리됨 	
트랜잭션 처리시간	상대적으로 오래걸림 (비트코인:10분, 이더리움:10초)	빠르게 트랜잭션 처리 (수초에서 실시간)	
유즈 케이스	디지털 화폐	은행 송금, 증권 거래 등 비즈니스 네트워크에서 사용	
구현 예	비트코인(Bitcoin), 이더리움(Ethereum)	리플(Ripple), 하이퍼레저(Hyperledger)	

업무사용의 중요성 : 비즈니스 유스케이스에 따라 분산합의 알고리즘을 목적에 맞게 사용할 수 있음

하이퍼레저 패브릭 합의 알고리즘

PBFT(Single)

- ✓ 하나의 거래당 합의과정을 통해서 블록이 생성됨

PBFT(Batch)

- ✓ 여러 트랜잭션을 정리하고 합의 과정을 거쳐 블록이 생성됨

Sieve (PBFT 알고리즘 개선)

- ✓ 체인코드 실행결과에 대한 합의정보를 얻은후에
- ✓ 각 노드의 체인코드 실행결과가 다른경우, 동기화 처리되지 않는 문제를 해결

* PBFT : Practical Byzantine Fault Tolerant

PBFT 처리흐름 구조

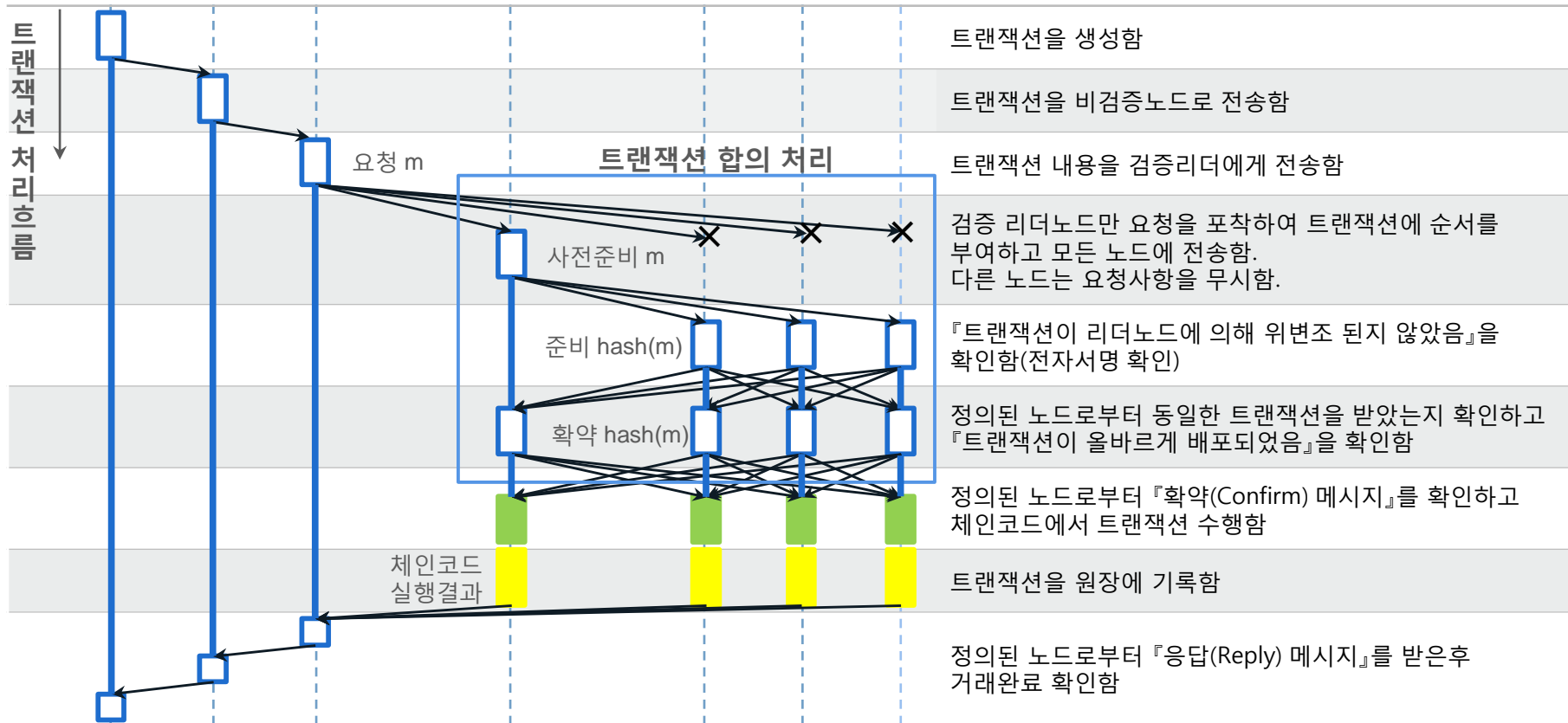
→ 통신 — 통신대기

m : 트랜잭션 메시지
hash(m) : 메시지 해쉬값

PBFT처리

체인코드실행

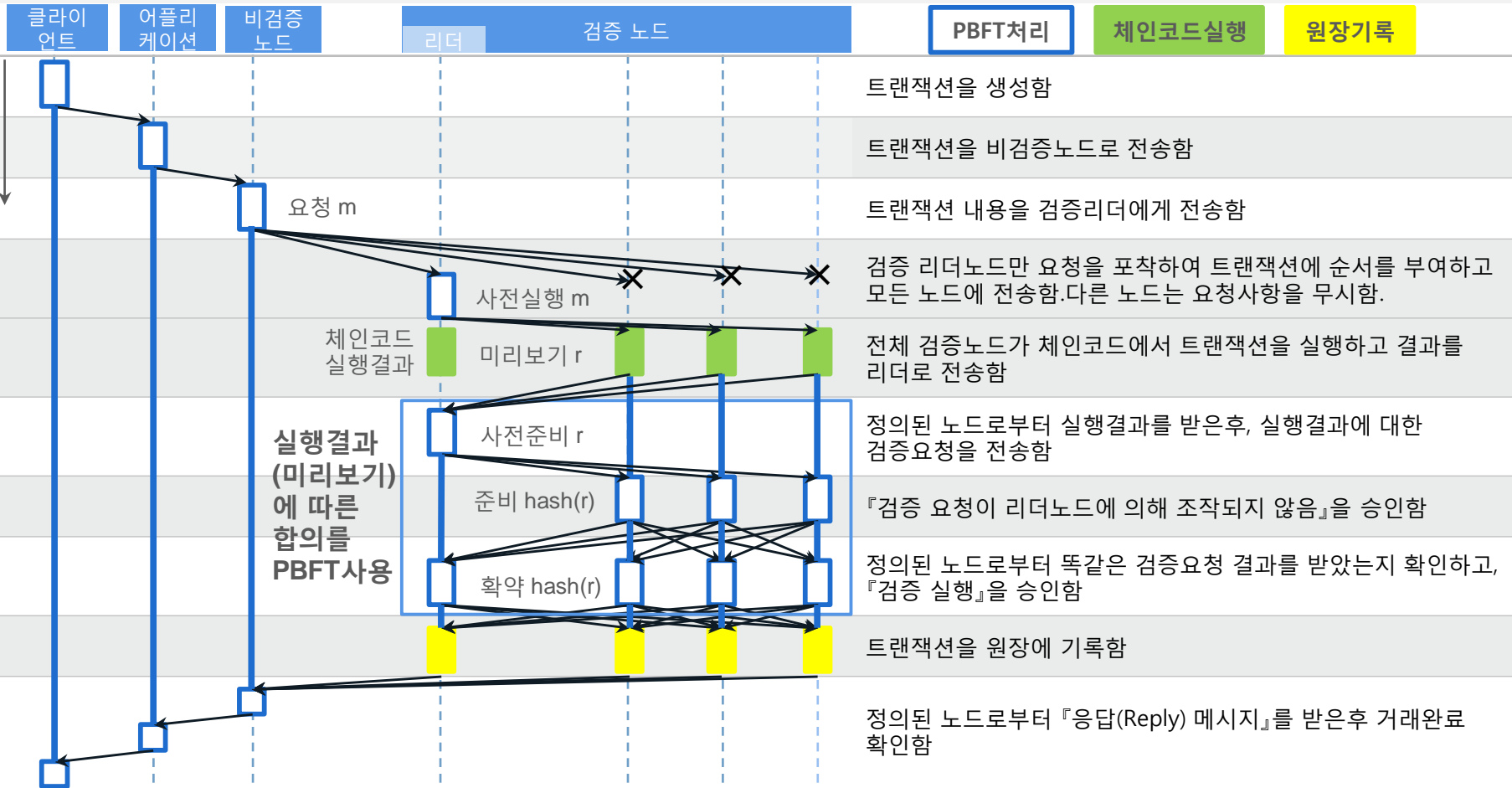
원장기록



Sieve 처리흐름 구조

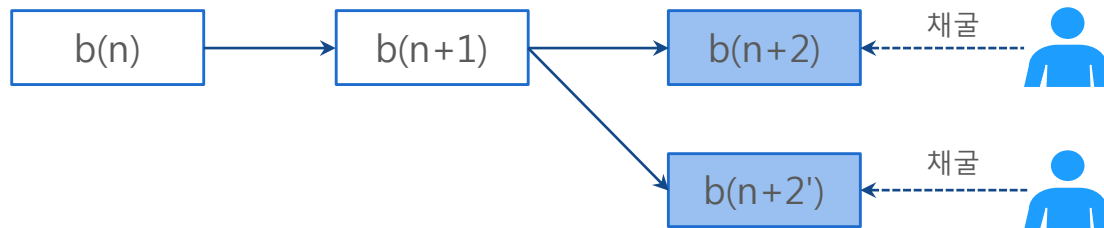
→ 통신 — 통신대기

m : 트랜잭션 메시지
r : 실행결과
hash(r) : 실행결과 해시값

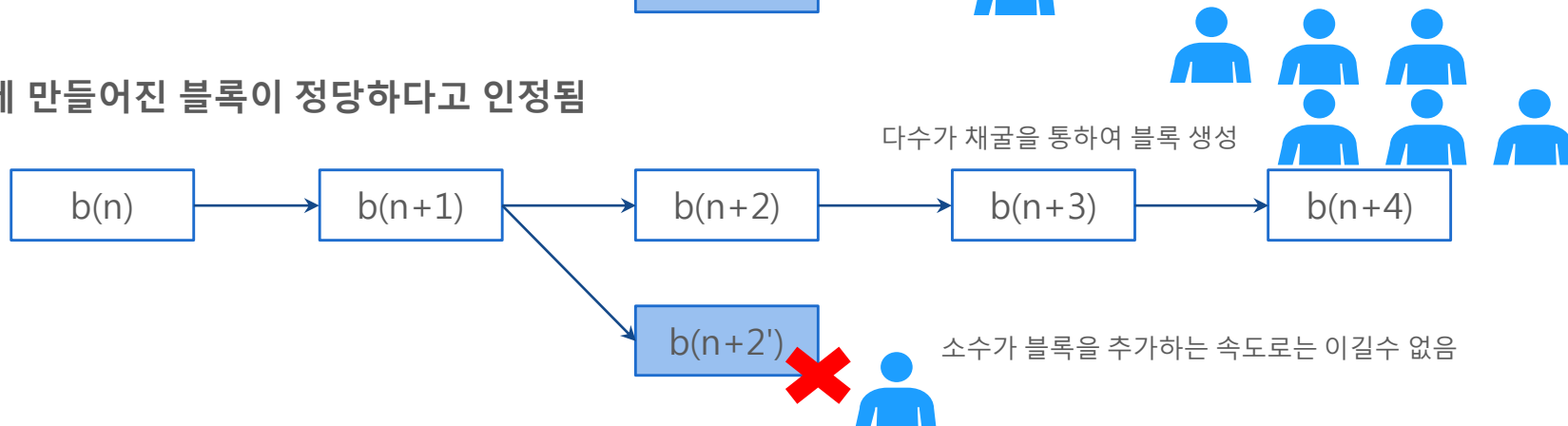


PoW(Proof of Work)의 문제점(1) > 최종성 부족

시점에 따라 여러 채굴(마이닝)이 거의 동시에 성공하여 블록이 분기하는 경우가 있음



길게 만들어진 블록이 정당하다고 인정됨



기각된 블록의 트랜잭션은 무효가 될 가능성이 있으며, 합의 알고리즘은 이러한 문제가 없어야 함

PoW(Proof of Work)의 문제점(2) > 51% 공격문제

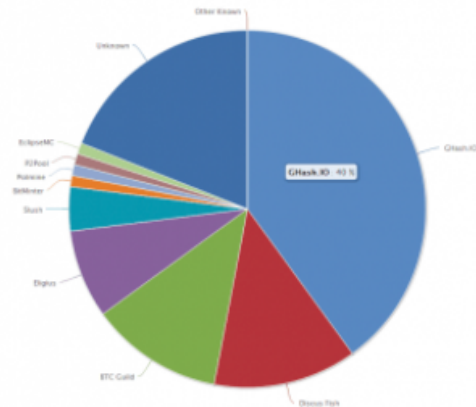
- ✓ 하나의 마이닝 풀(또는 개인의 부)이 51%이상의 컴퓨팅 파워를 가진경우 블록체인 변조가 가능함
- ✓ 2014년 6월경 Ghash.io의 채굴 속도(hash rate)가 51%에 접근했다고 화제가 되었음
 - Bitcoin 가격이 폭락함

Bitcoin Mining Pool Ghash.io Is Unapologetic Over Risk Of Theoretical 51% Attack

👤 Caleb Chen 📁 Altcoin Mining, Bitcoin Mining, News 💬 17 Comments



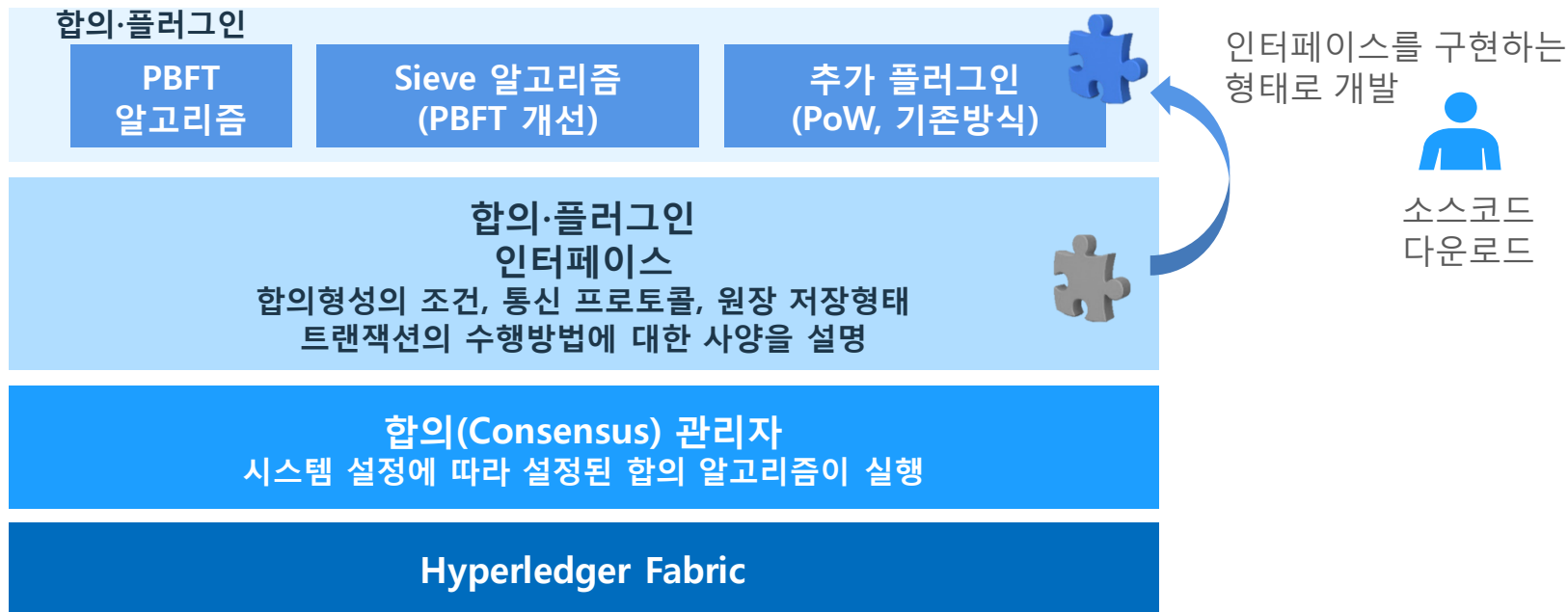
Update: [Ghash.io has responded](#) as expected. Two Cornell researchers that have been trying to poke holes in Bitcoin's network security, [Ittay Eyal](#) and [Emin Gün Sirer](#), have taken this opportunity to remind the Bitcoin community of many other theoretical attacks that can occur with a centralized Bitcoin mining network. In fact, many of the theoretical attacks described would arguably be more likely than a blatant 51% attack. However, **any attack**, no matter how subtle,



Many are uncomfortable with Ghash.io's large percentage of total network hashrate

플러그인 구조

하이퍼레저 패브릭(Hyperledger Fabric)은 다양한 업무에 적용될수 있도록 PBFT 합의 알고리즘에 한정하지 않고, 임의의 합의 알고리즘을 실현할 수있는 플러그인 구조를 채용하고 있음. 하이퍼레저를 이용하는 기술자는 업무 이용에 따른 합의 알고리즘을 실현하는 추가 플러그인을 개발하고 하이퍼레저에 통합 할 수 있음.



보안 및 프라이버시

비즈니스
네트워크내에 모든
거래가 기록되고
공유됨

Shared
ledger
(공유원장)

원장은 공유되지만,
참여자의 개인정보는
암호화 기술을 통해서
보호되어야 함

Security &
Privacy
(보안&프라이버시)

Smart
Contract
(스마트 계약)

비즈니스 규칙 및
로직은 계약에
함축되어 트랜잭션
수행시 실행됨

Consensus
(합의)

검증된 트랜잭션에
대한 네트워크에
참여한 참여자의
동의가 필요함

... 참여확대, 비용절감, 효율성 증대

기업용 블록체인을 위한 보안 요구사항

신원 관리(Identity)

- 업무 사용을 위해서는 참가자의 신원 확인이 필요
 - ✓ 책임 스푸핑(Spoofing) 방지
- 참가자의 신원인증 (참가자의 정체성과 특성의 확인)

트랜잭션의 기밀성(Confidentiality)

- 트랜잭션이 암호화되어 일반 사용자에게 보이지 않도록 함

재생(Replay) 공격 대책

- 과거의 트랜잭션을 복사하여 재전송하는 공격을 방지함

개인정보 보호

- 트랜잭션의 발행자의 익명화
- 동일한 사용자가 발행한 여러 트랜잭션들은 연관성이 없도록 함

액세스 제어

- 스마트계약을 실행할 수 있는 사람을 제한 (초기화, 함수 실행, 데이터 참조 등)

PKI 인증서와 전자서명

- 사용자의 신원을 확인함
- 개별 트랜잭션은 익명의 인증서를 통해 신원을 숨긴 채 인증 할 수 있음

IBM의 연구보고에 따르면, 기업용 블록체인은 허가형 블록체인 (Permissioned Blockchain)이 유일한 방법이라는 결론

인증서

인증서(certificate)는 사용자 non-validating peer, validating peer 등 모든 엔터티에 발행됨

Ecert(Enrollment Certificate)

- ✓ 사용자의 신원을 확인하기 위해서 장기적으로 사용되는 인증서
- ✓ 사용자, 검증노드 및 비검증 노드에 발행됨

Tcert(Transaction Certificate)

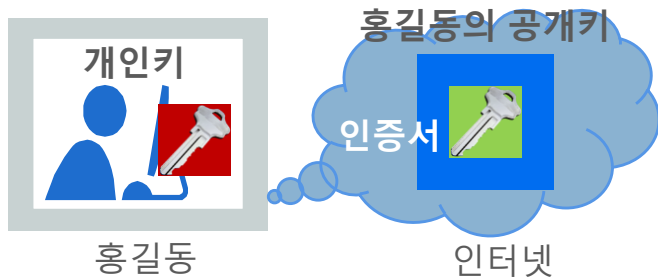
- ✓ 트랜잭션마다 발행되는 단기적으로 사용되는 인증서
- ✓ 사용자, 검증노드 및 비검증 노드에게 발행됨
- ✓ 트랜잭션마다 다른 Tcert를 사용하면 여러 트랜잭션이 동일한 사용자에게 속하는 것을 숨길수 있음

TLS 인증서(Certificate) : 통신 채널에 암호화하는 데 사용되는 인증서

공개키(Public key) 암호화 방식

공개키(Public Key)와 개인키(Private Key) 쌍

- ✓ 공개키는 비밀이 아니며 모두에게 공유되어도 문제가 없음
- ✓ 개인키는 키 소유자만 보유함



개인키로 암호화 -> 공개키로 복호화

- ✓ 개인키의 소유자가 보낸 메시지임을 확인 할수 있음 (예:서명)



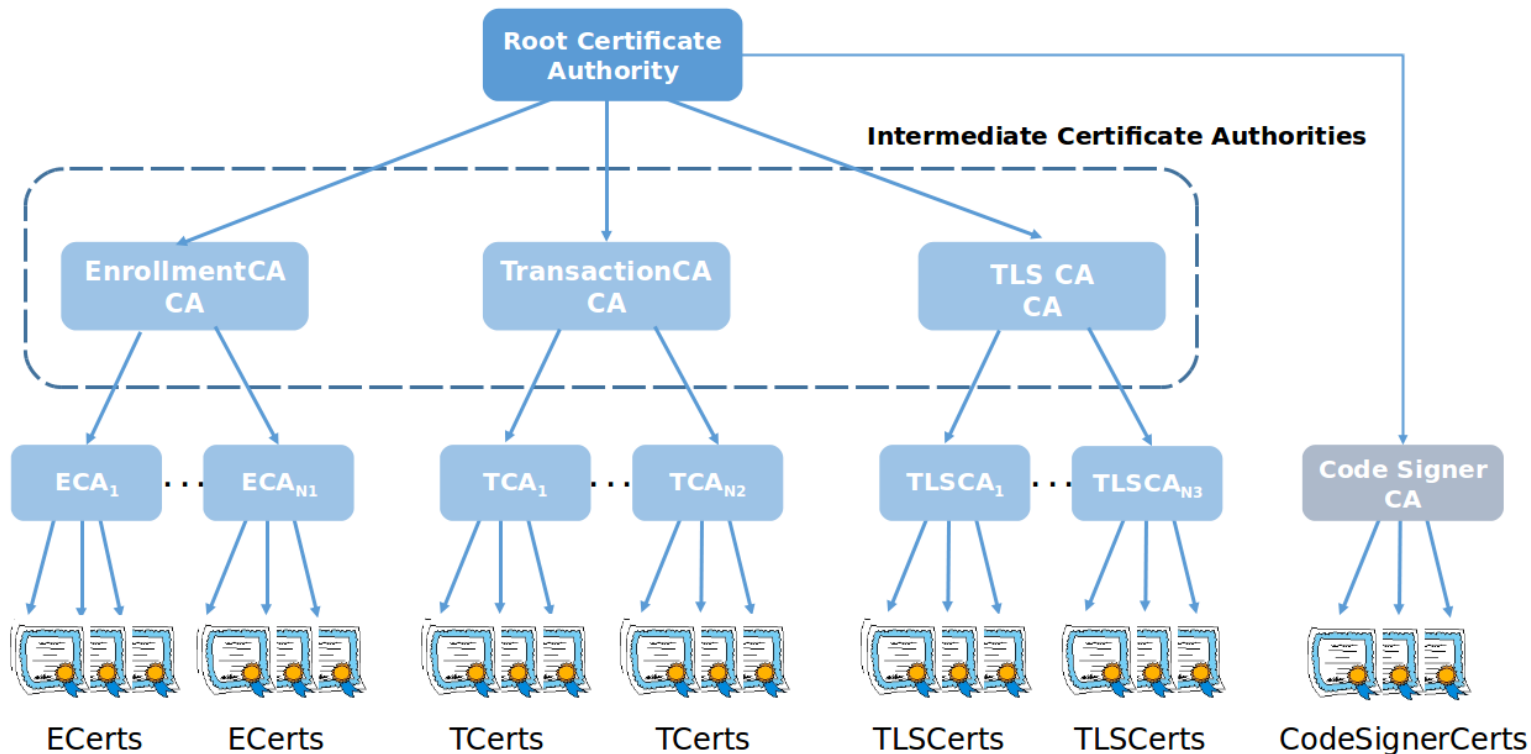
공개키로 암호화 -> 개인키로만 복호화 가능

- ✓ 개인키의 소유자만이 읽을 수있는 개인메시지를 보낼 수 있음



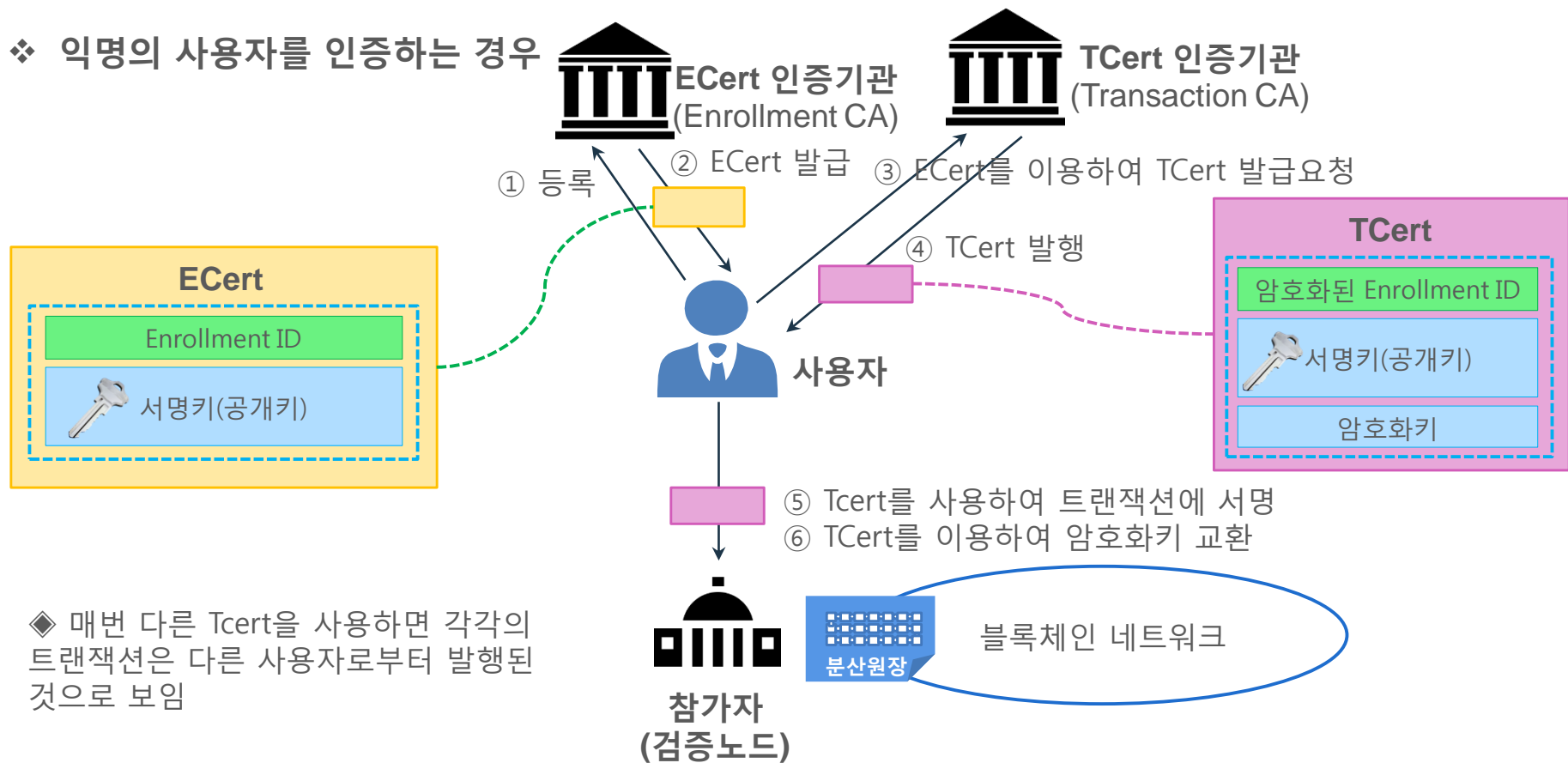
하이퍼레저의 PKI 계층구조

Public Key Infrastructure - Hierarchy



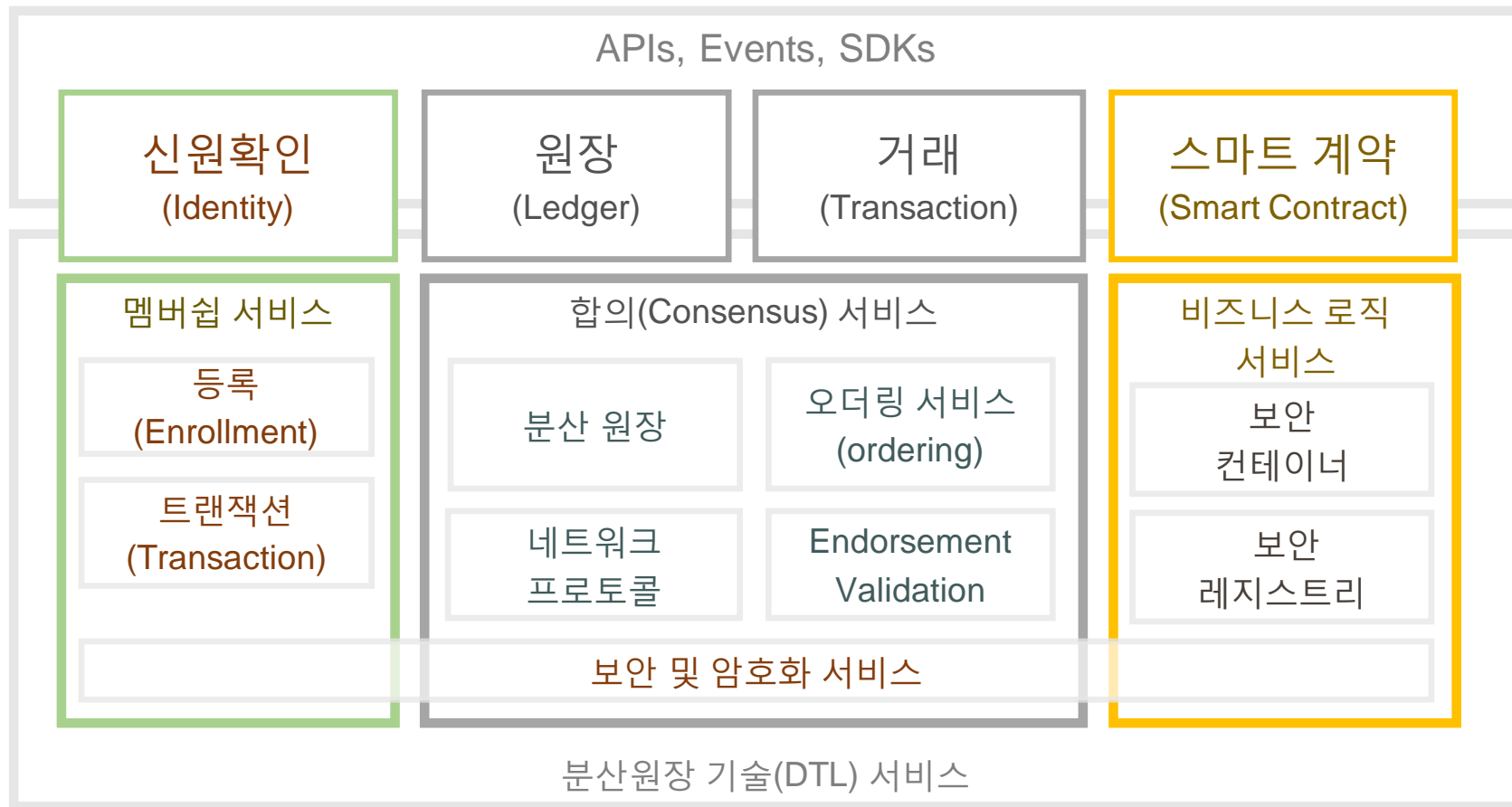
인증서 사용 모델

❖ 익명의 사용자를 인증하는 경우



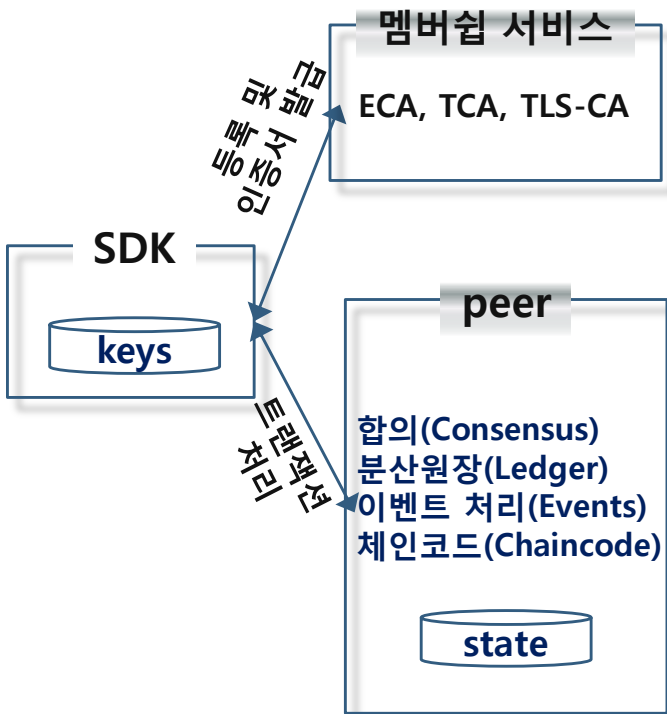
◆ 매번 다른 Tcert을 사용하면 각각의 트랜잭션은 다른 사용자로부터 발행된 것으로 보임

하이퍼레저 패브릭 V1.0 아키텍처

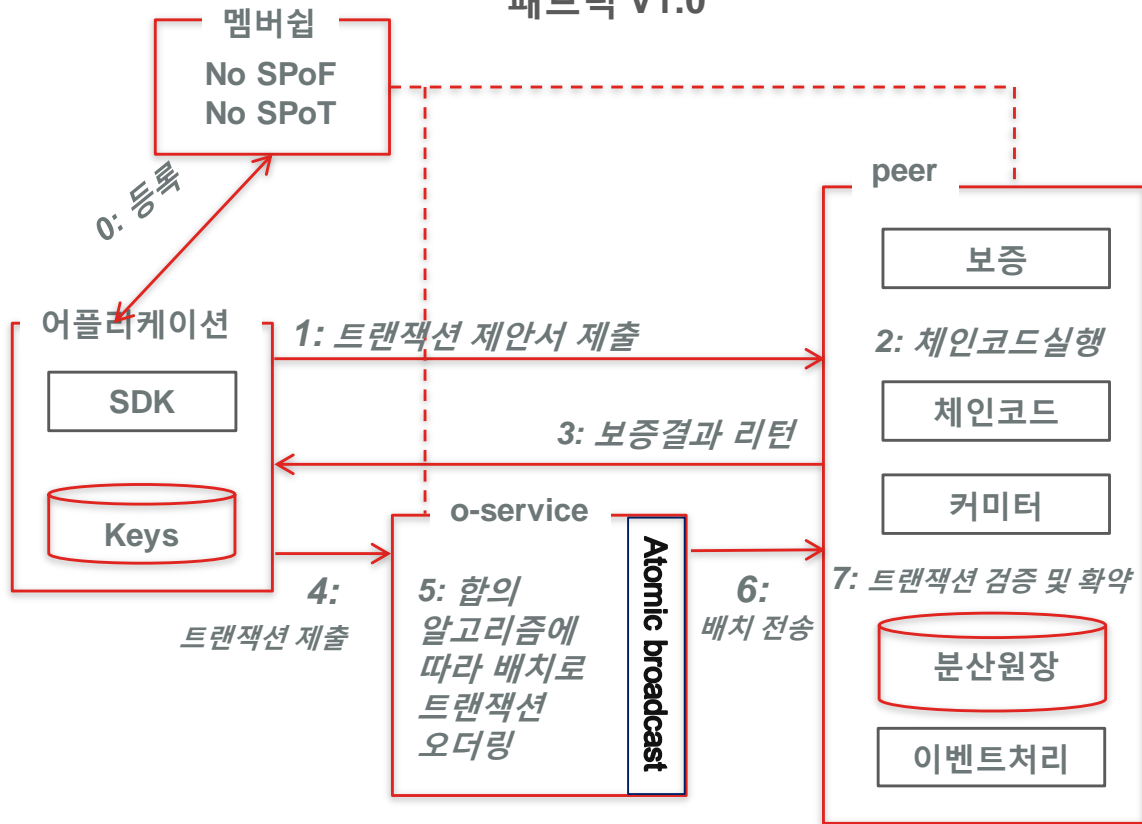


하이퍼레저 패브릭 V0.6과 V1.0 비교

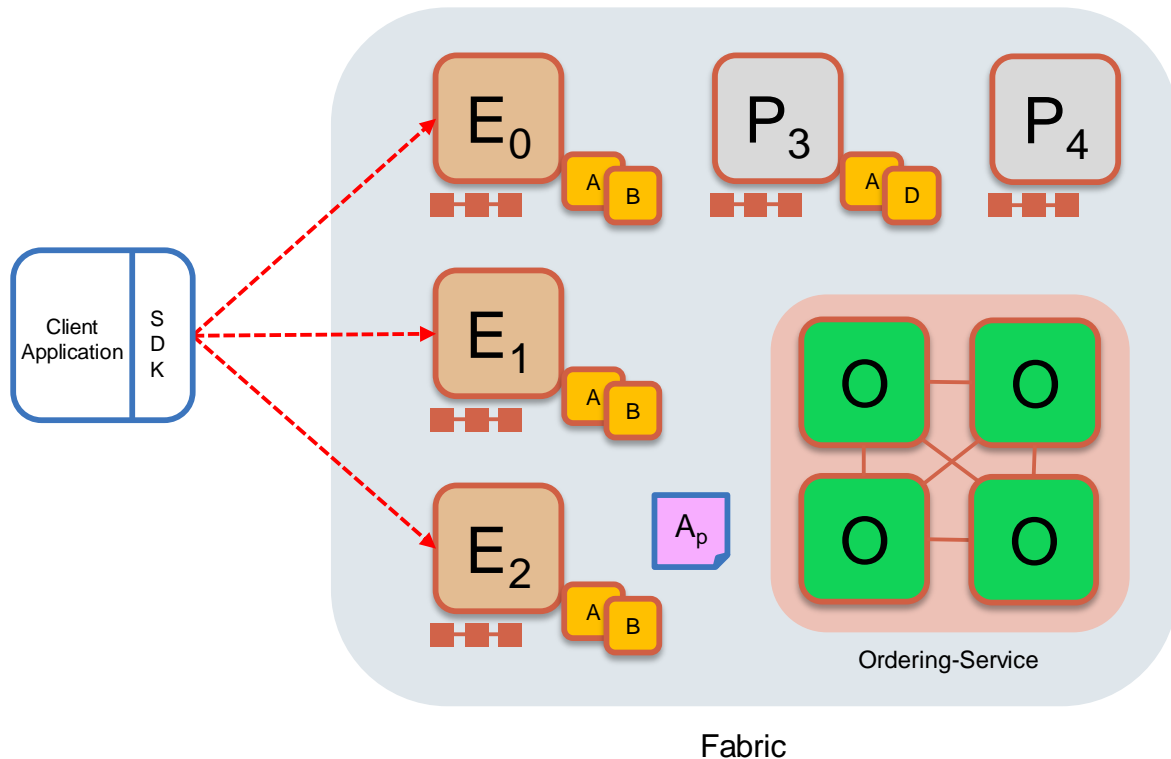
패브릭 V0.6



패브릭 V1.0



트랜잭션 실행 예제 > ①거래 제안



클라이언트의 거래 제안

보증 정책:

- “ E_0 , E_1 및 E_2 는 반드시 전자서명”
- (P_3 , P_4 노드는 정책의 대상이 아님)

클라이언트 어플리케이션은 체인코드 A 실행을 위한 거래 제안을 하게되며, 반드시 $\{E_0, E_1, E_2\}$ 노드들이 대상이 됨

Key:

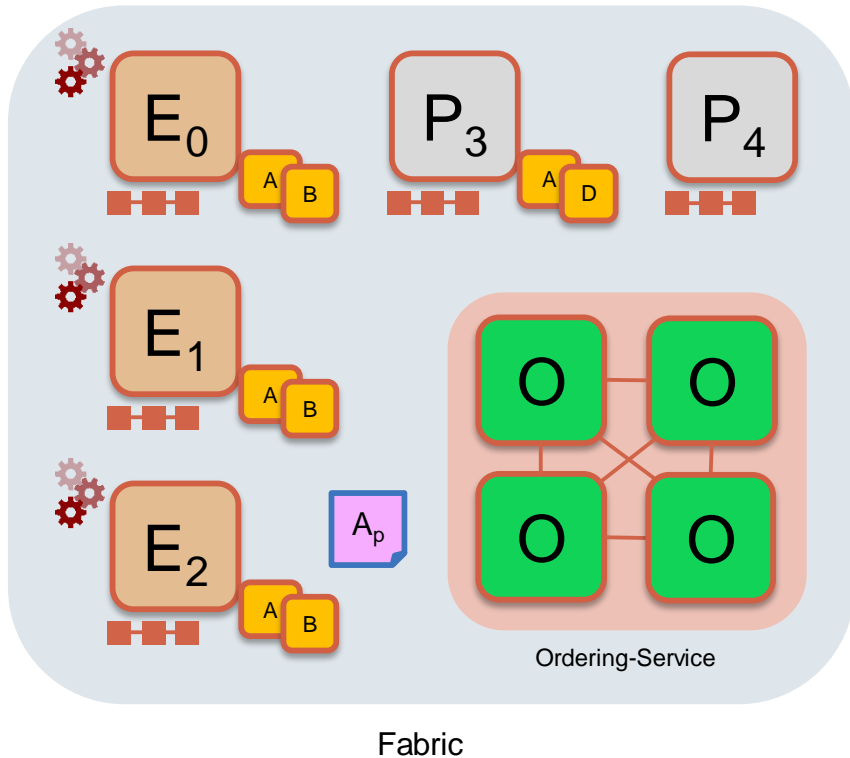
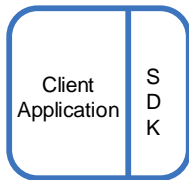
Endorser			Ledger
Committer			Application
Orderer			
Smart Contract (Chain code)			Endorsement Policy

트랜잭션 실행 예제 > ②거래제안 실행

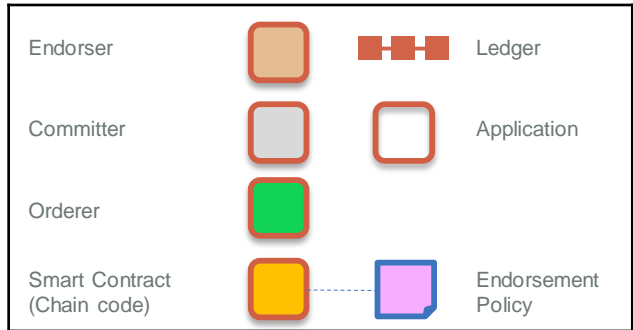
보증 노드의 거래제안 실행

E_0 , E_1 및 E_2 는 각각 제안된 거래를 실행하지만 원장에 실행결과를 업데이트하지 않음

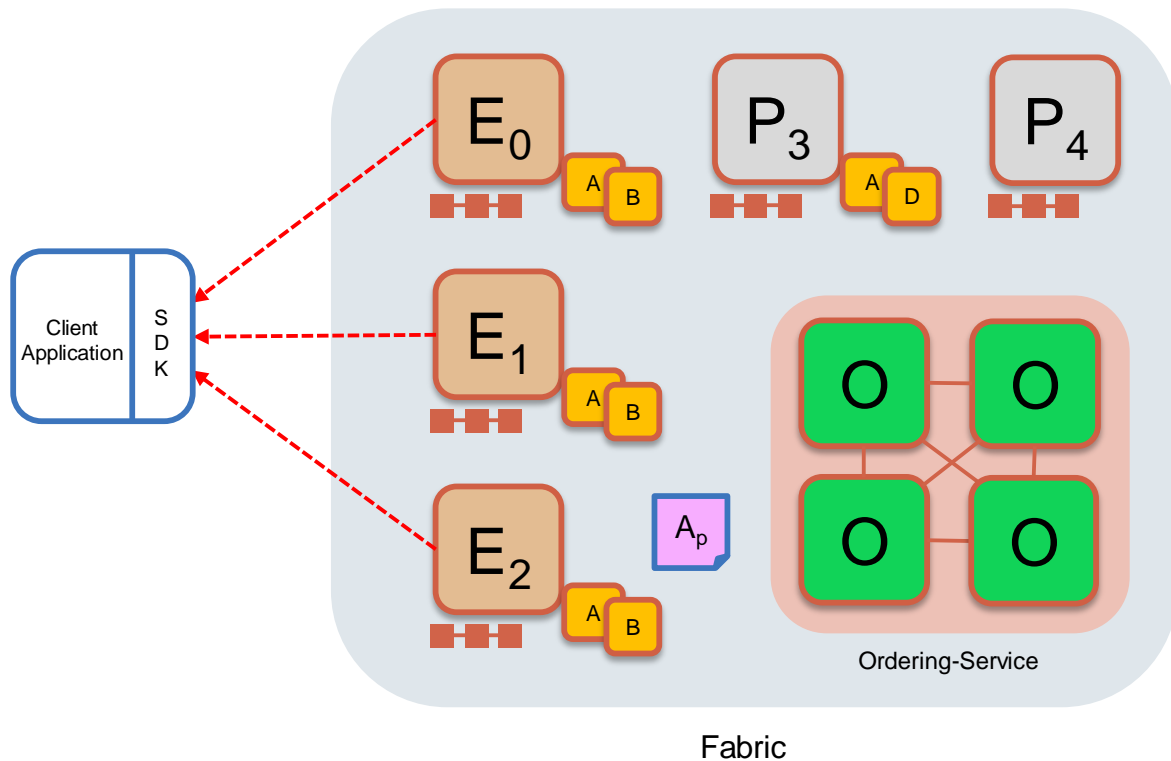
각 실행을 통해서 “RW 세트”인 읽기(read), 쓰기(write) 데이터를 캡처하여 패브릭내에서 전송됨.



Key:



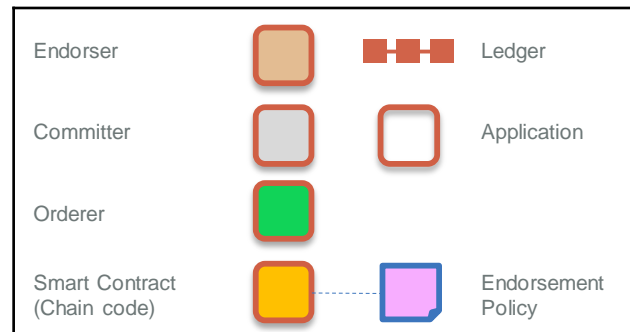
트랜잭션 실행 예제 > ③ 거래제안 응답



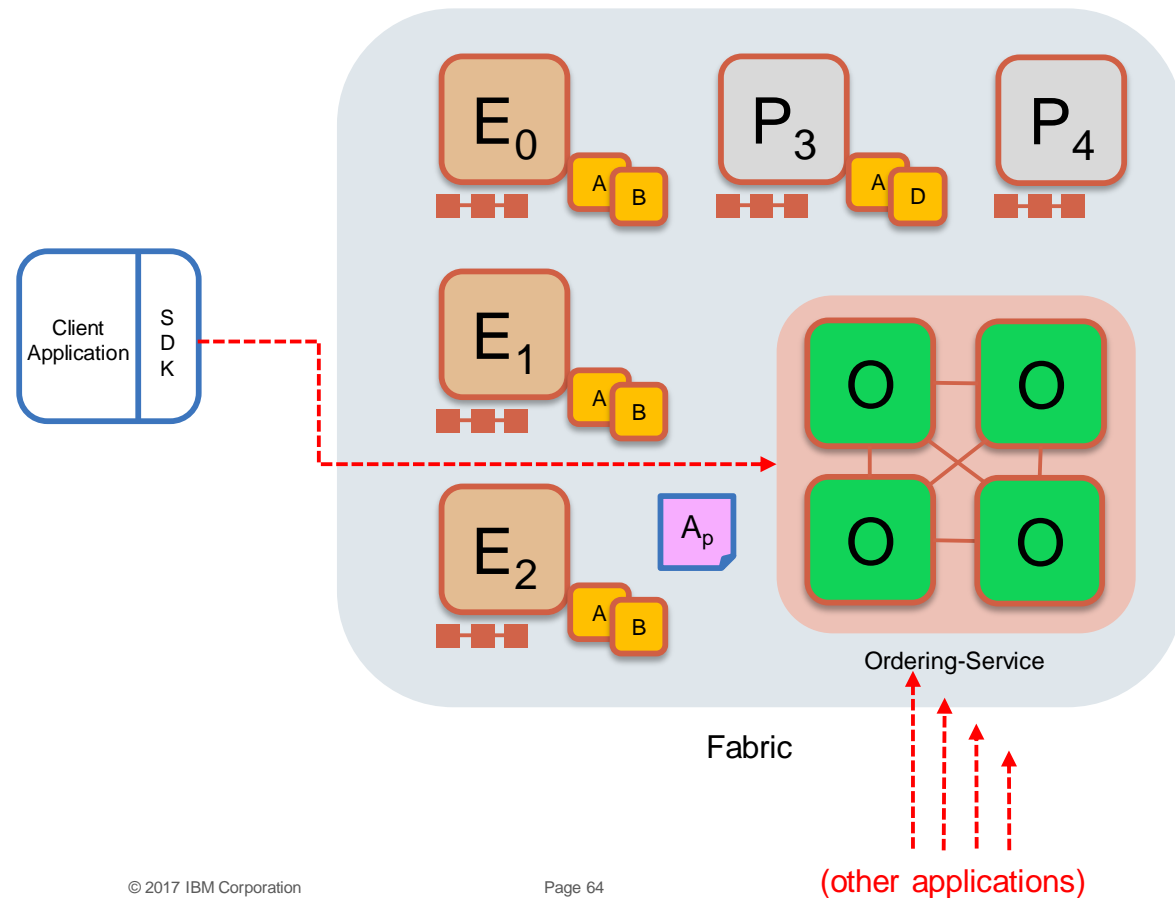
클라이언트는 응답 수신

각 보증 노드로부터 서명된 RW 세트를 서명후에 클라이언트에게 결과값을 송신함

Key:



트랜잭션 실행 예제 > ④ 거래 오더링

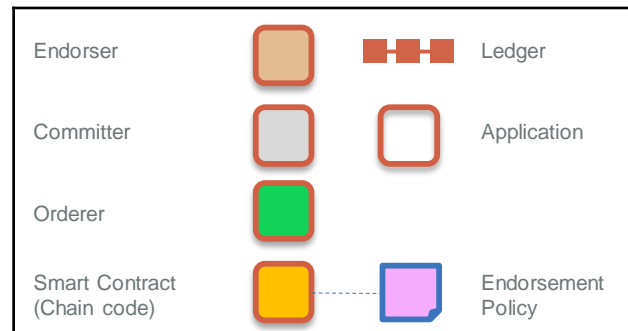


**클라이언트는 오더링 서비스에 보증
노드로부터 받은 결과 제출**

클라이언트는 보증 피어로부터 받은
응답을 오더링할 거래로 제출함

오더링은 다른 클라이언트
어플리케이션에서 제출된 거래들이
동시에 패브릭내에서 처리됨

Key:



트랜잭션 실행 예제 > ⑤ 거래 전달

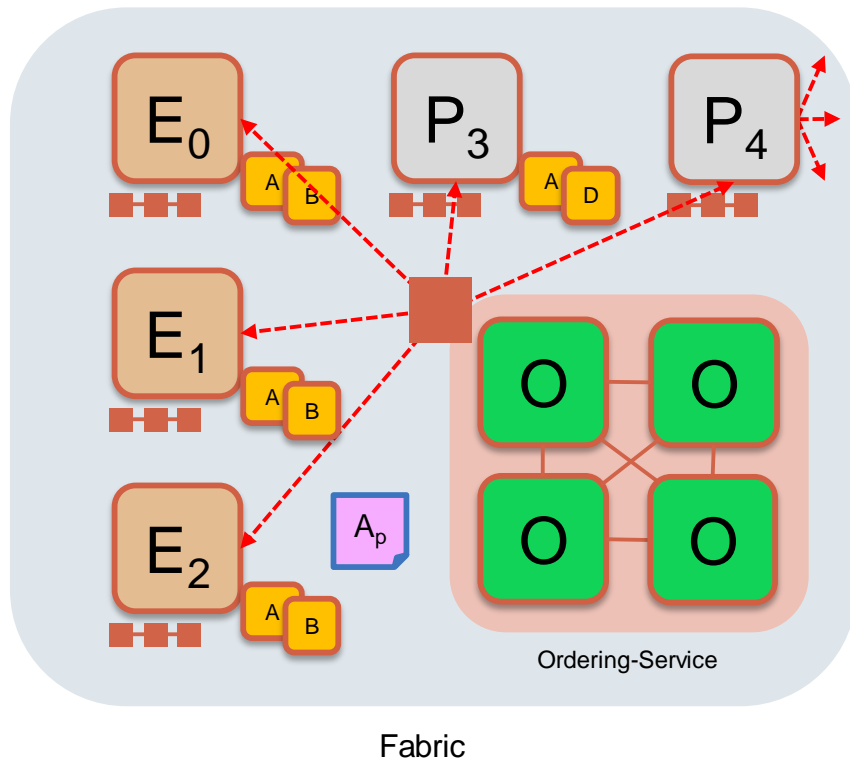
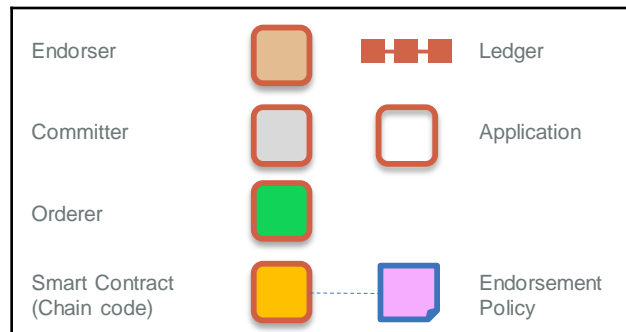
오더러는 모든 커미터 피어에 전달

오더링 서비스는 거래를 수집하여 블록을 생성하여 커미터 피어들에 전달함. 피어들은 가십(gossip) 프로토콜을 사용하여 다른 피어들에게 전달함(P4가 해당)

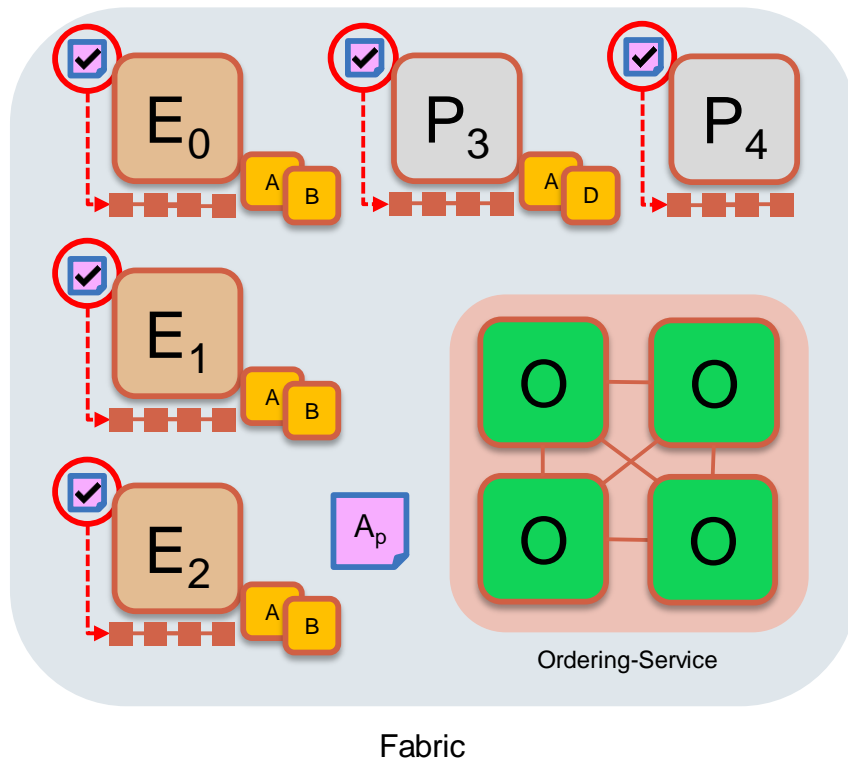
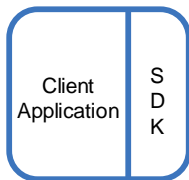
다른종류의 오더링 알고리즘 제공:

- SOLO (single node, 개발용)
- Kafka (블록을 토픽으로 매핑)
- SBFT (PBFT와 유사)

Key:



트랜잭션 실행 예제 > ⑥ 거래 검증

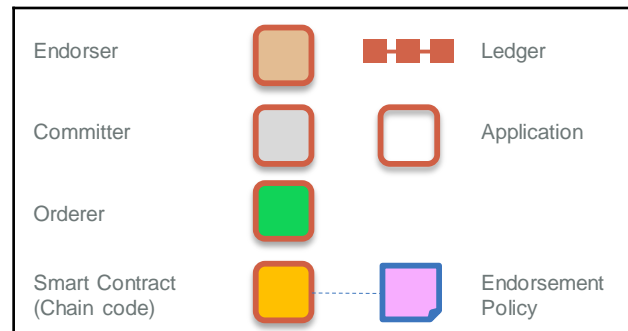


커미터 피어들은 거래를 검증

모든 커미터 피어들은 보증 정책에 따라 거래를 검증하며, 현재 상태에서 RW 세트가 유효한지 점검함

거래는 원장에 기록되고, 검증된 거래로 캐싱 DB가 변경됨

Key:

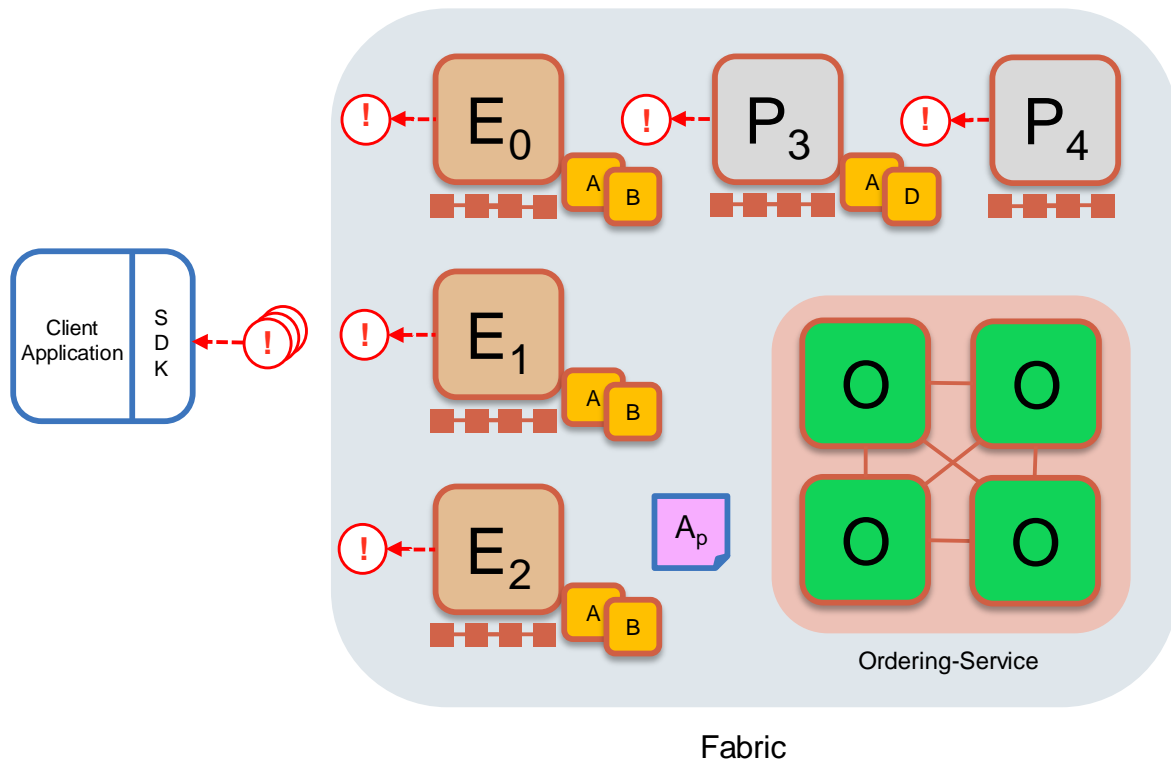


트랜잭션 실행 예제 > ⑦ 거래 결과통보

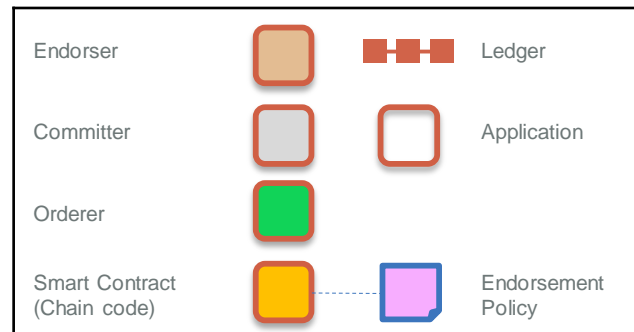
커미터 피어들은 클라이언트에게 결과 통보

거래가 성공 또는 실패되었을 때, 블록이 원장에 추가되었을 때, 클라이언트는 통보를 받도록 등록될 수 있음

클라이언트와 연결된 각 피어들을 통해 결과 통보를 받을 수 있음

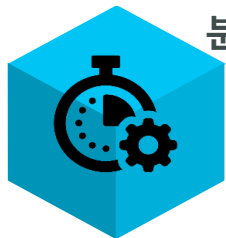


Key:



하이퍼레저 패브릭 V1.0

기밀성



분할 실행

체인코드 실행과 트랜잭션 오더링을 분리함으로써 네트워크 성능을 최적화 시킴



허가형 멤버십

알려진 참여자 및 규제감독을 기반으로 신뢰된 블록체인 네트워크 운영



멀티 채널

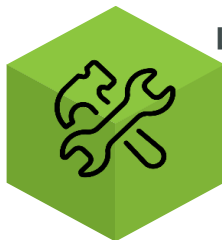
규제대상 산업에 필요한 개인정보 보호 및 기밀 유지기능등을 통해서 다자간 거래에 필요한 멀티 서비스 제공

운영 워크로드



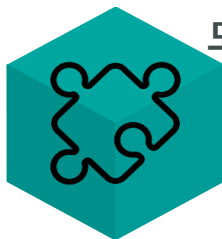
거래 내역

효율적인 감사 및 분쟁해결을 위한 검색 가능한 거래내역 조회



네트워크 도구

IBM은 모니터링, 로깅 및 컴플라이언스를 위한 백업/복원 도구를 제공

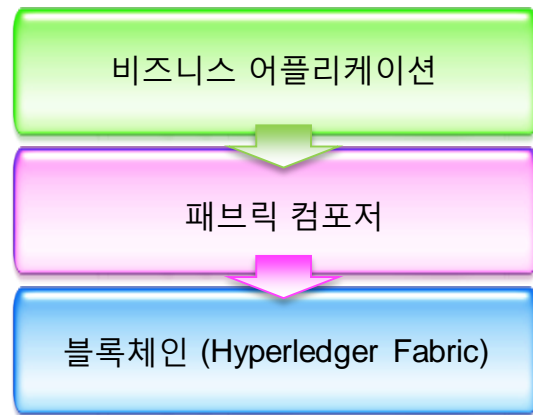


모듈러 아키텍처

비즈니스 네트워크를 동적으로 확장 용이한 노드 수, 합의 알고리즘, ID 관리, 암호화에 대한 환경 설정

블록체인 비즈니스 네트워크 모델링 도구

- 블록체인 비즈니스 네트워크를 위한 하이-레벨 어플리케이션 추상화 스위트(suite)
- 신속하게 솔루션 생성을 위한 비즈니스 중심의 용어 집중
- 리스크를 줄이고, 이해와 유연성을 높임



```
asset Vehicle identified by vin {
  o String vin
  --> Member owner
}

participant Member identified by email {
  o String email
  o String firstName
  o String lastName
  o Double balance
}

transaction Offer identified by transactionId {
  o String transactionId
  o Double bidPrice
  --> VehicleListing listing
  --> Member member
}
```

- 기능
 - 비즈니스 네트워크를 모델링하고, 테스트 및 APIs를 공개
 - 응용 어플리케이션은 비즈니스 네트워크와 통신하기 위해 APIs 트랜잭션을 호출
 - Loopback/REST를 이용하여 기존시스템과 통합
- 오픈 소스로 <http://fabric-composer.org> 사이트에서 활용 가능

Contents



What is Blockchain?



Why is it relevant
for our business?



How can IBM help
us apply Blockchain?



First Projects
Use cases

하이퍼레저 패브릭 기반의 First 프로젝트(1/2)

FX Netting



Settlements through digital currency



Diamond Provenance



Identity management



Credit Defaults Swaps



Trade Finance



Channel Financing



Low liquidity securities trading and settlement



Bike blockchain



Reward points management



Food Safety



Contract Management



Global Trade Digitization



Trade Logistics



Health Data Exchange



환율고시 및 외화송금



하이퍼레저 패브릭 기반의 First 프로젝트(2/2)

Private Equity 	Supply chain finance 	Bond Trading 	Asset custody system 
Finance, real estate and asset custody 	Managing customer's digital identities 	Commodity trade finance for US crude oil transactions 	Supply chain financial services for pharmaceutical procurement 
Corporation identity management 	Cotton trading consortium 	Property portfolio revaluation 	Logistics / Supply chain 
Digital trade finance instrument 	Know Your Customer 	Blockchain experiment 	Loyalty and Rewards Platform 

블록체인을 활용한 식품안전망



요구사항

- ✓ 원자재, 기계류(NSF 인증) 및 가공처리(HACCP 인증), 물류(검역소독 인증)에 관한 **주요정보의 안전한 문서화**
- ✓ 규제기관 및 소비자를 위한 전자 바코드나 QR 코드를 통한 **정보 라벨링이 부착된 제품패키지 생성**
- ✓ 제품을 패키지로 수집
- ✓ 제품패키지의 분해 및 판매점을 위한 재포장 작업
- ✓ 부정한 대체물이나 위조품의 유출을 막기 위한 패키지 검증작업
- ✓ 운송라인 전반의 제품패키지의 가시성 확보
- ✓ 모바일 앱을 통한 정보추적검색 (사육사부터 소비자까지의 유통과정을 추적)
- ✓ 계산대에서 만료된 제품(또는 알람 발생)에 대한 거래 확인
- ✓ 제품 출처와 관련하여 고객의 평가 및 선호도
- ✓ 공급망 전반의 각 제품 유통과정에 대한 **엔드-투-엔드 가시성**
- ✓ 기계류, 만료 날짜, 제품 수량 등과 관련된 **규제준수** 점검
- ✓ 공급망 관리 **인증** 및 **감사 레코드** 생성

블록체인 기반의
자산 추적
(Provenance)
기능

블록체인
데이터와
라벨링과의
연관성

제품 데이터
수집 및 분해

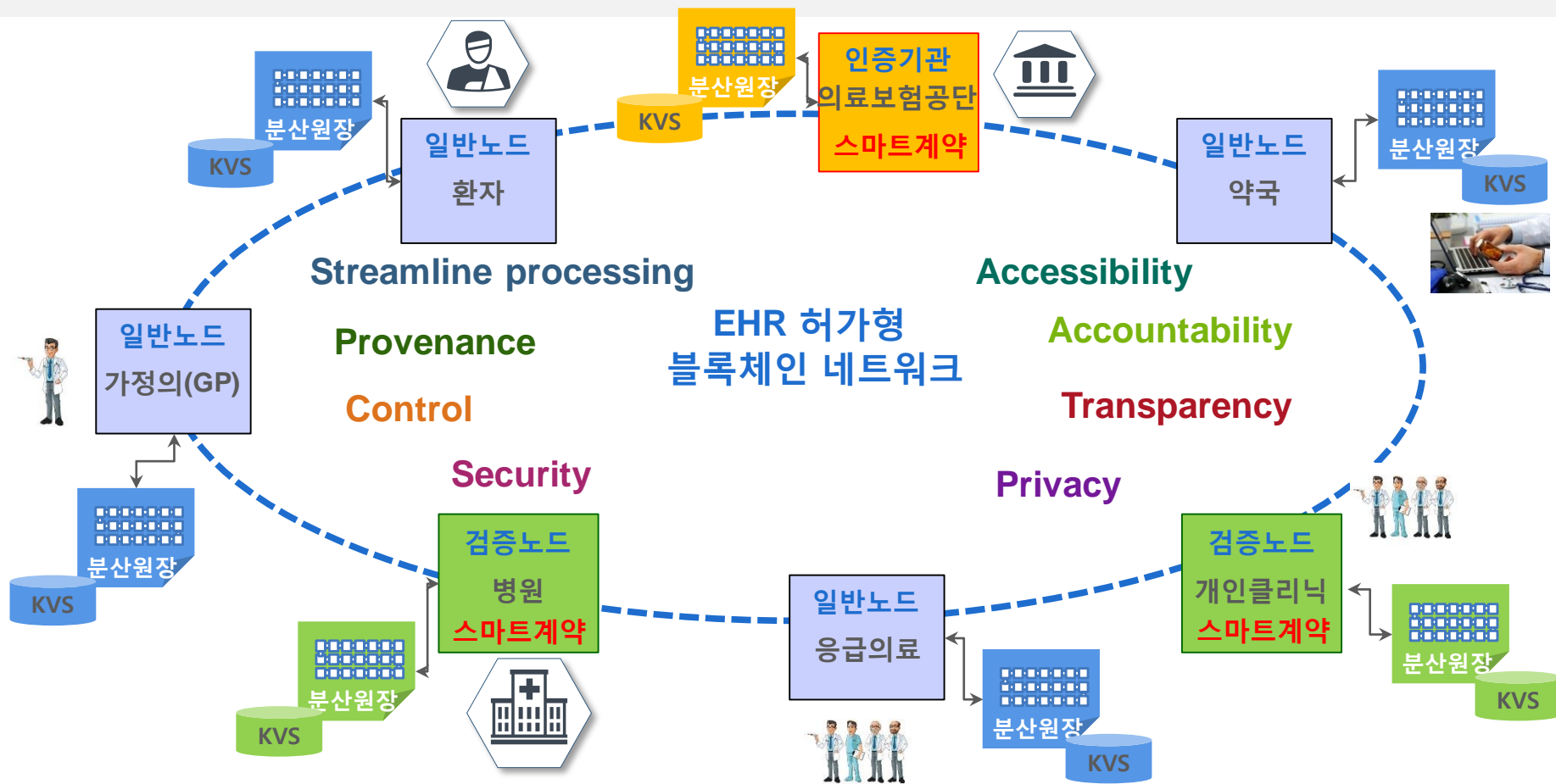
엔드-투-엔트
허가형 가시성

불변성을
기반으로 한 진품
추적성

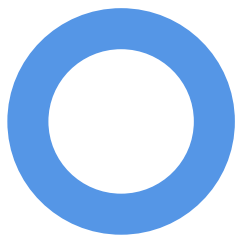
간단한
방법으로의
정보검색

감사 기능 :
스마트계약을 통한
규제변화에 대한 대응

EHR을 위한 블록체인 네트워크



블록체인 적용이 적합한 사례

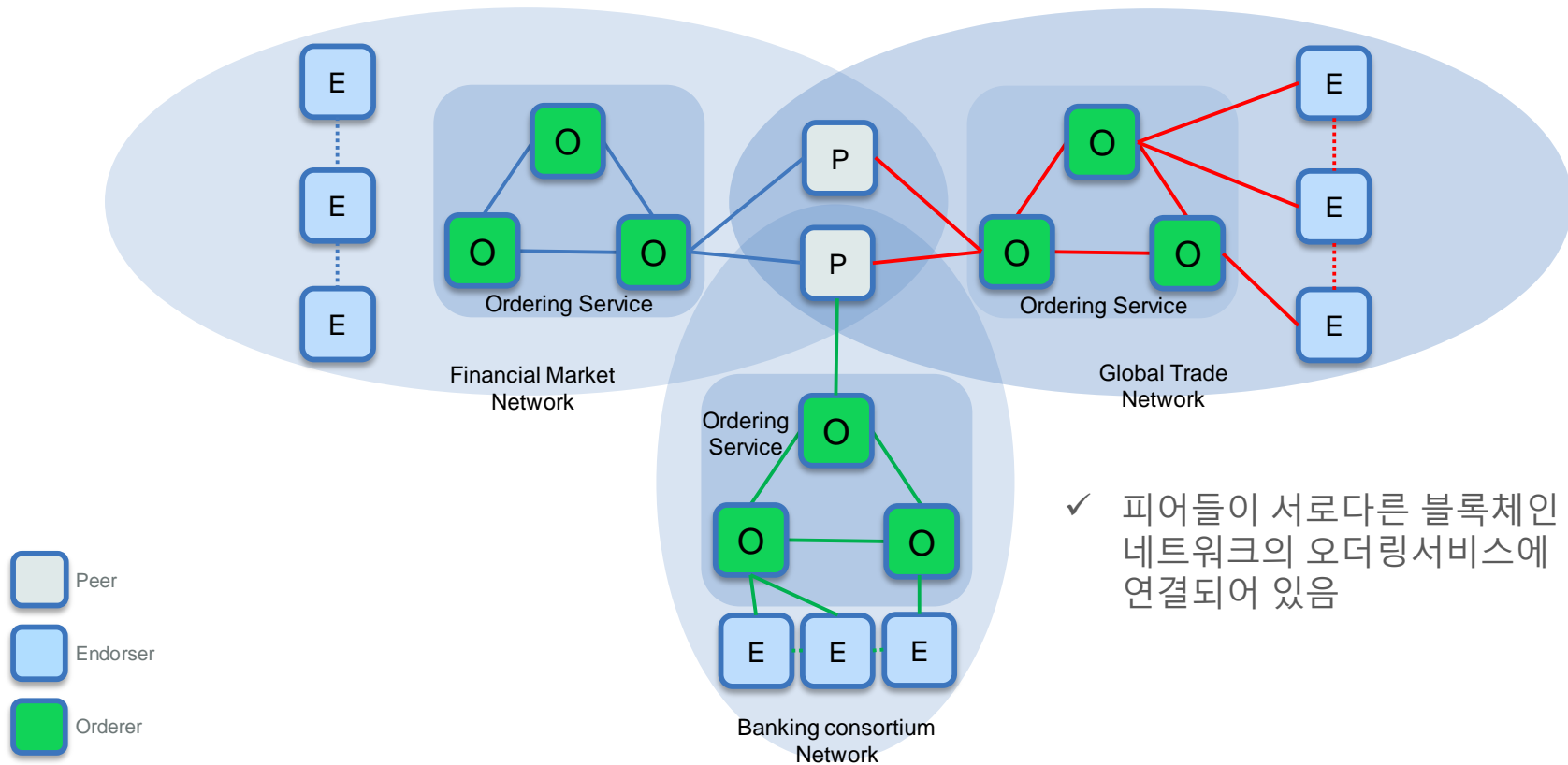


- ✓ 분산 원장 및 처리에 의해 공동시스템 사용이 불필요 함으로, **비용 절감 효과**가 기대됨
- ✓ 일부 노드가 정지하고 있어도 처리가 계속 있기 때문에 **안정한 시스템을 구축** 할 수 있음
- ✓ 각 노드 (각 참가자) 사이에서 동일한 비즈니스 로직, 데이터 공유, 위변조할 수 없는 상태에서 거래 기록이 유지되기 때문에 가능 **감사 기능을 향상**됨



- ✓ 분산 원장의 합의를 수행하면서 갱신되기 때문에 현재 고성능 (높은 응답)이 요구되는 시스템
- ✓ 조직 및 회사를 포함하지 않고 하나의 조직만 참여하는 경우 (일반적으로 시스템에 대응 가능)
- ✓ 간단한 DB / 트랜잭션 처리의 대체

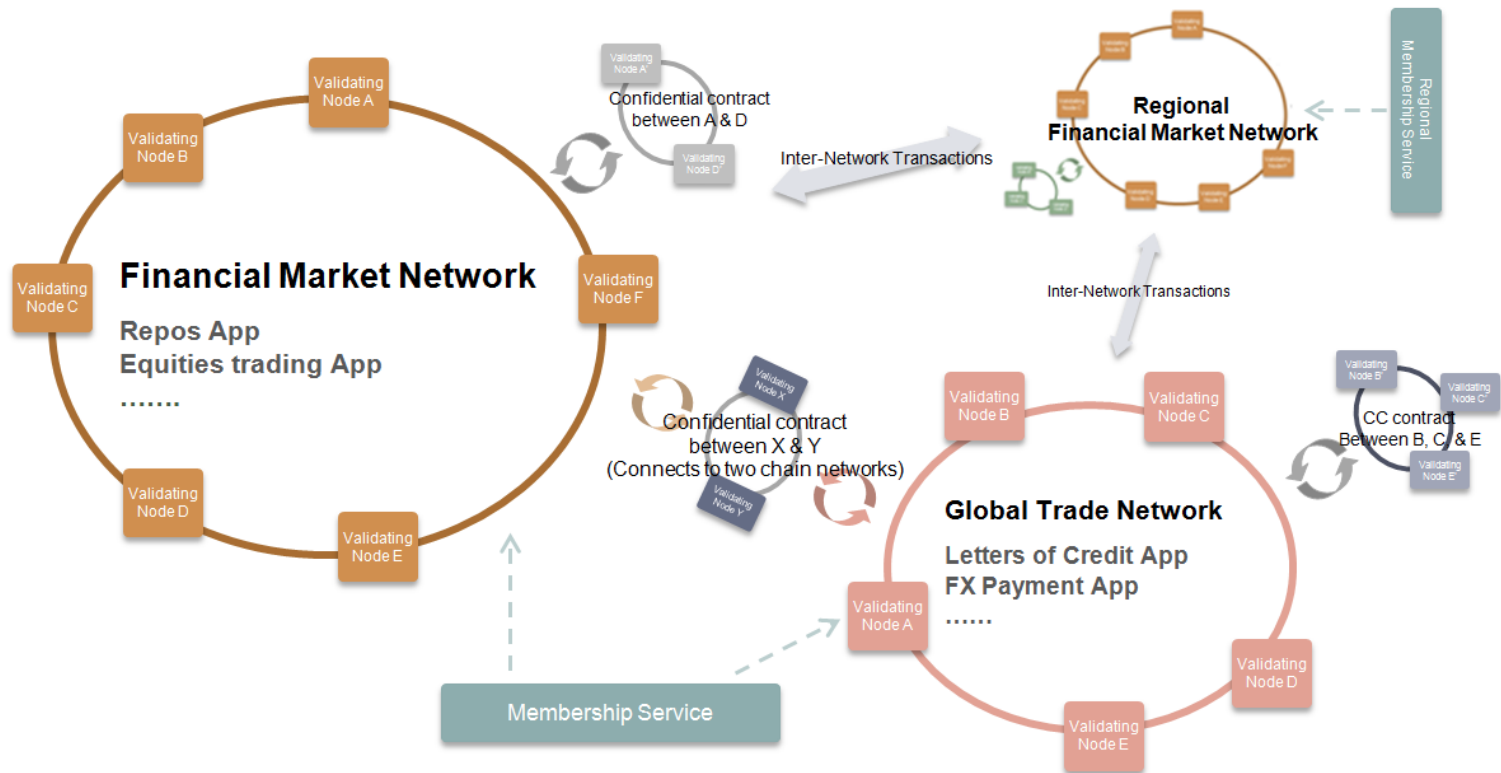
미래의 블록체인 네트워크 형태



✓ 피어들이 서로다른 블록체인 네트워크의 오더링서비스에 연결되어 있음

블록체인의 미래

서로 다른 비즈니스 목적을 가진 블록체인 네트워크가 생길 것으로 예상



Why Hyperledger Fabric

산업용 표준
블록체인 기술

표준화
기술

- 1 비영리 단체인 리눅스 재단에서 운영하고 있는 블록체인 오픈소스 프로젝트로 130여개의 회원사가 가입하여 운영되고 있음
- 2 기업들의 다양한 비즈니스 영역에 사용될 수 있는 허가형 (permissioned) 블록체인 플랫폼
- 3 하이퍼레저 프로젝트 회원인 R3CEV*도 Corda를 하이퍼레저 프로젝트에 오픈 소스로 2016년 11월 30일 기부

차별화된
블록체인
아키텍처 설계

뛰어난
아키텍처

- 4 산업 표준에 따른 오픈소스 기반으로 사용자 정의 스마트 컨트랙 (Smart Contract), 강력한 암호화 지원 및 인증(Identity) 기능 제공
- 5 플러그블(pluggable) 분산형 합의(consensus)을 지원하는 모듈러(modular) 아키텍처로 설계되었으며, 20년이상 검증된 PBFT (Practical Byzantine Fault Tolerance) 알고리즘 제공등 V1.0에서는 합의 알고리즘의 재정의의 통해 1000 tps 이상의 성능이 지원받는 아키텍처 제공

블록체인 기술의
빠른변화 및
업무연속성
확보를 위한
지원 체계

거버넌스

- 6 하이퍼레저 프로젝트의 거버넌스를 기반으로 개방적이고 투명한 개발 프로세스를 통하여 향후 로드맵을 지원하고 가속화
- 7 IBM은 OBC(Open BlockChain)를 하이퍼레저 프로젝트에 기부하였으며, 하이퍼레저를 기반으로 100개 이상의 글로벌 기관들과 First 프로젝트 수행경험이 많음

경청해 주셔서 감사합니다!



박 세열
블록체인 기술리더/
금융총괄 아키텍트

Blockchain Technical Leader/
Client Technical Leader
S&D Architect Team
Technical Sales
Banking & Financial Markets



IBM Korea
Three IFC, 10 Gukjegeumyung-ro,
Yeongdeungpo-gu, Seoul, Korea

Mobile: +82-10-4995-7163
Phone: +82-2-3781-7163

E-mail: sypark@kr.ibm.com

