ORACLE

ORACLE

Oracle Blockchain Platform 소개

Wonjo Yoo
Principal Solution Engineer
Cloud Excellence Team, Oracle Korea
Oct 22, 2019

Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, timing, and pricing of any features or functionality described for Oracle's prod ucts may change and remains at the sole discretion of Oracle Corporation.

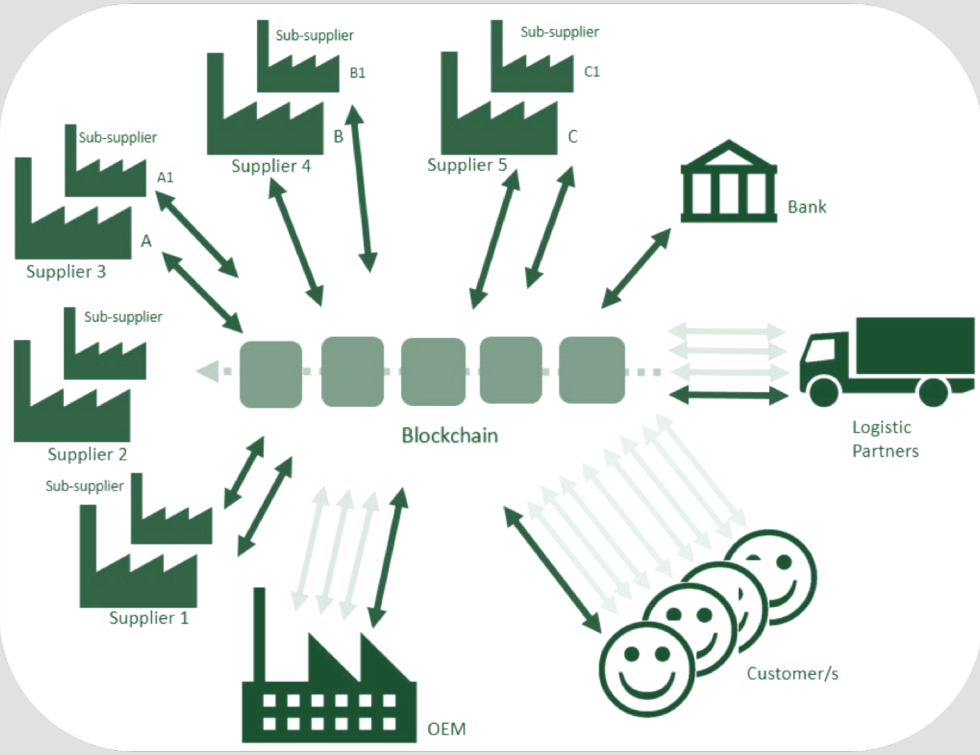
Program agenda

1. Hyperledger Fabric
2. Oracle Blockchain Platform

ORACLE

Hyperledger Fabric

블록체인의 5가지 특징



- 준 실시간 업데이트, 합의
- 중개자 대체 : 암호화로 peer간의 신뢰
- 신뢰할 수 있는 공유와 투명한 데이터 접근
- 신뢰할 수 있는 거래와 부인방지
- history 를 통한 트랜잭션 조작 방지

블록체인의 두가지 유형

허가형 Permissioned

- 허가된 멤버만이 네트워크에 참여
- 예: 제조업체 및 공급 업체,화물 운송 업체 및 브로커 네트워크 등
- 보다 안전하고 향상된 관리 – 사전에 알려진 참가자
- **기업형으로 적합함**

비허가 (Open)

- 인터넷에 있는 누구나 가입하여 원장 정보에 액세스 할 수 있음
- 예: Bitcoin, Ethereum, 등
- 덜 안전하고 빈약한 관리 - 사전에 알려지지 않은 참가자

Hyperledger Fabric

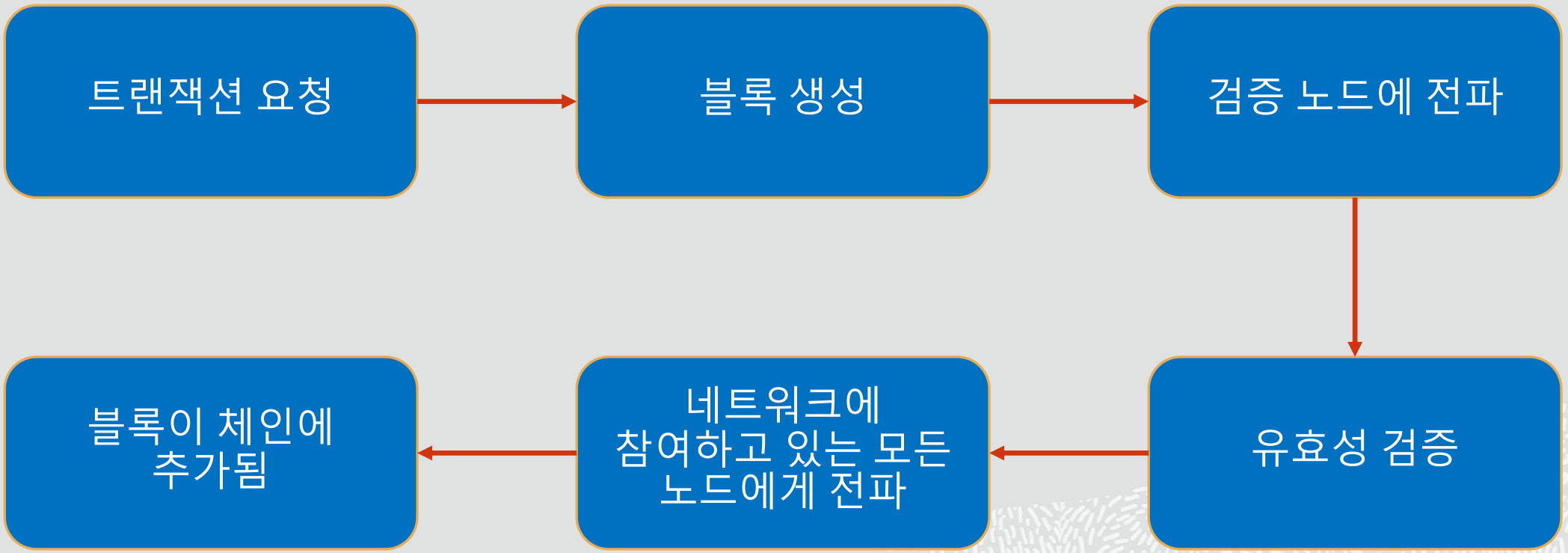
Open source enterprise-grade permissioned distributed ledger technology (DLT) platform

https://www.hyperledger.org/

https://hyperledger-fabric.readthedocs.io/en/release-1.4/index.html

https://www.youtube.com/watch?v=EKa5Gh9whgU

기본 동작 방식

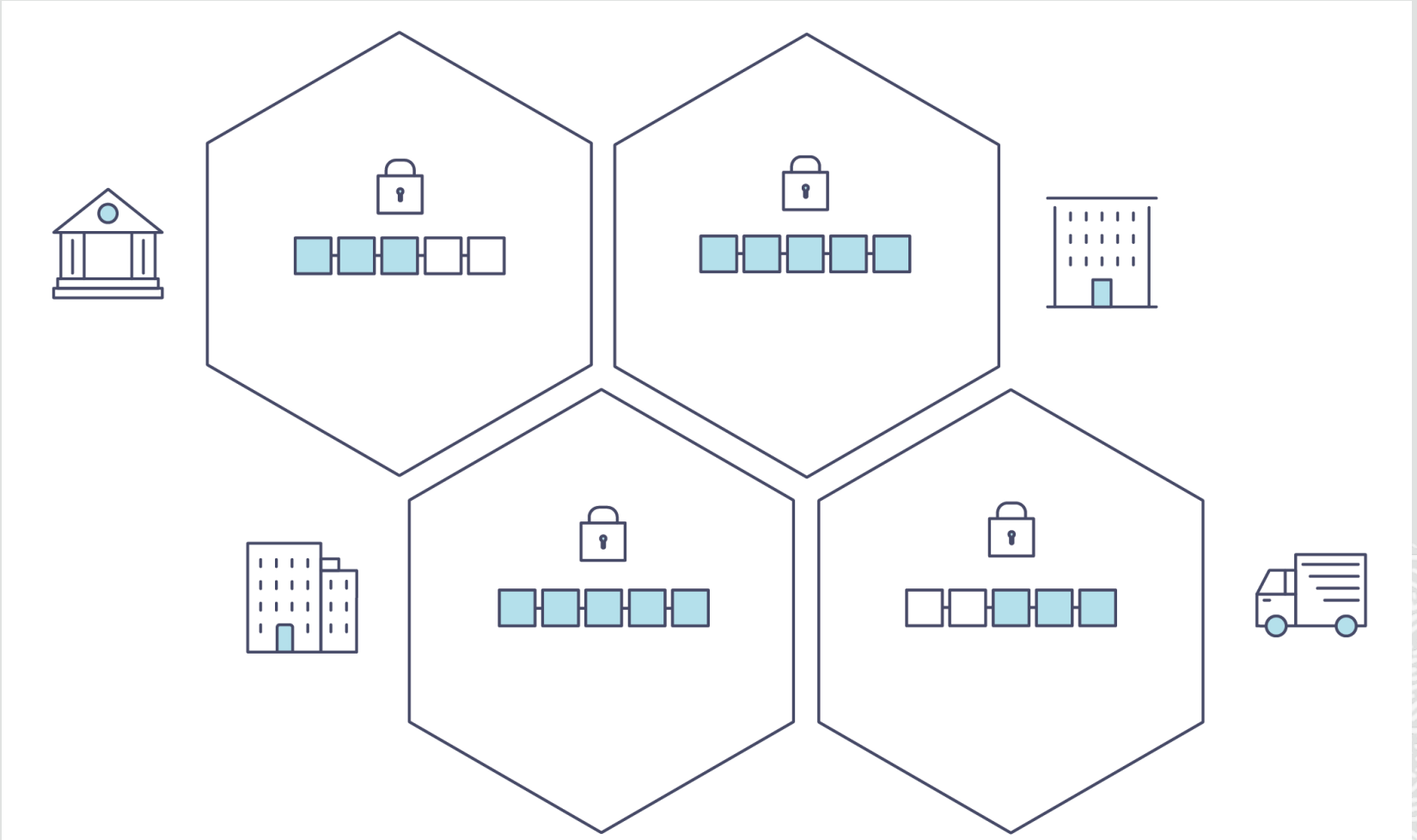

트랜잭션 요청
블록 생성
검증 노드에 전파
유효성 검증
네트워크에
참여하고 있는 모든
노드에게 전파
블록이 체인에
추가됨

Hyperledger Fabric – Basic Concepts

- Distributed Ledger (분산 원장)
- Smart Contracts (계약서)
- Consensus (합의)

Basic Concepts – Distributed Ledger (분산 원장)

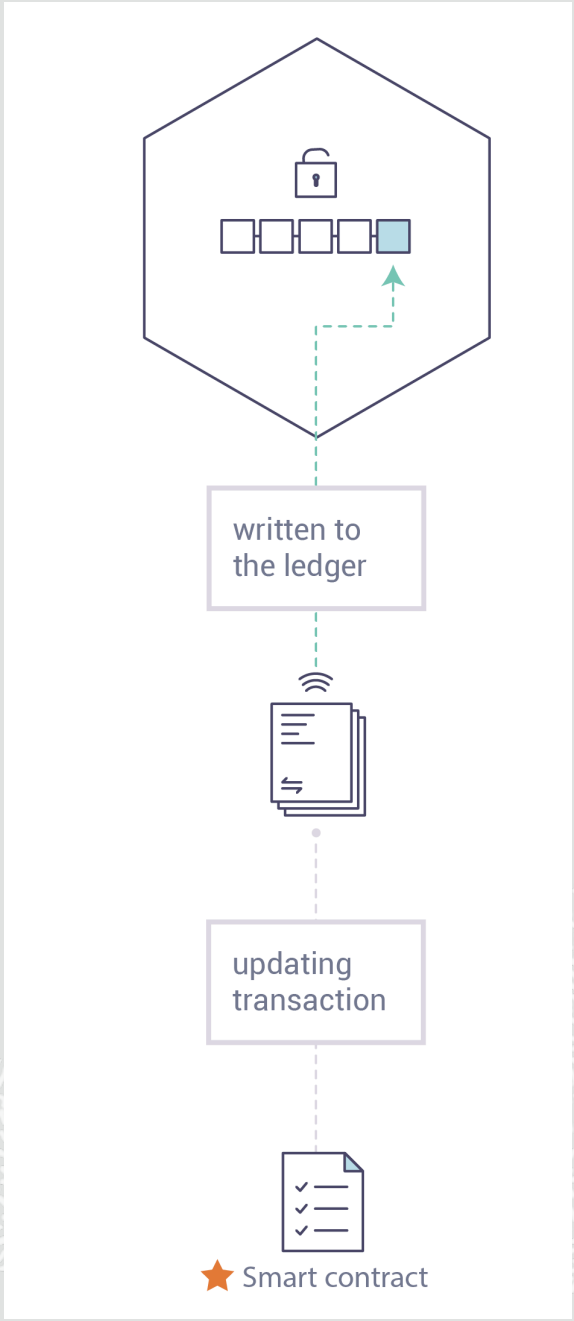
Distributed Ledger



Basic Concepts – Smart Contract (계약서)

Smart Contracts : 계약서

현재 Go, node, java 지원



Basic Concepts – Smart Contract (chaincode)

Seller Organization

ORG1

```
application:

seller = ORG1;
buyer = ORG2;
transfer(CAR1, seller, buyer);
```

```
car contract:

 query(car):
  get(car);
  return car;

 transfer(car, buyer, seller):
  get(car);
  car.owner = buyer;
  put(car);
  return car;

 update(car, properties):
  get(car);
  car.colour = properties.colour;
  put(car);
  return car;
```

Buyer Organization

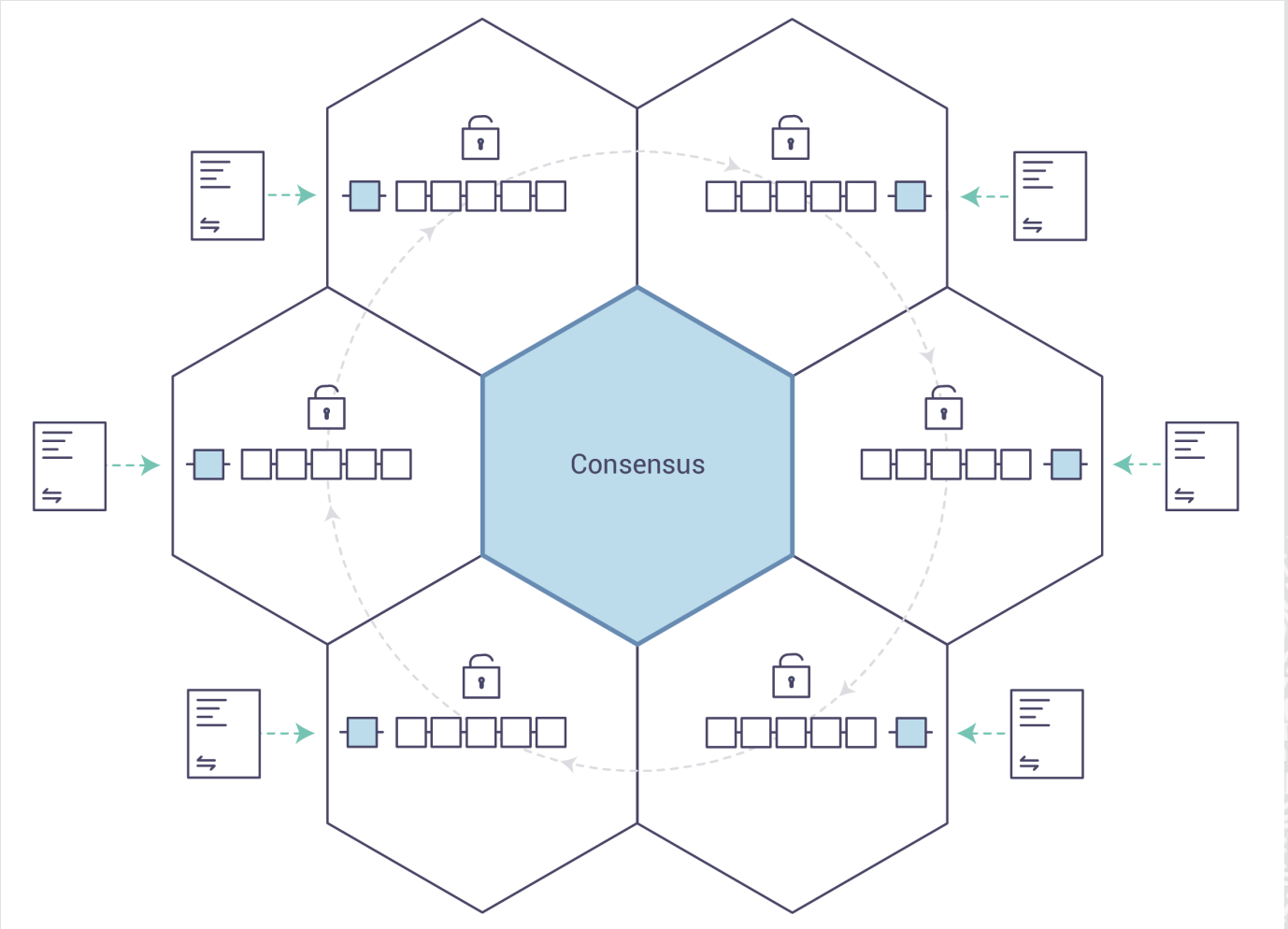
ORG2

```
application:

seller = ORG2;
buyer = ORG1;
transfer(CAR2, seller, buyer);
```

Basic Concepts – Consensus(합의)

Consensus



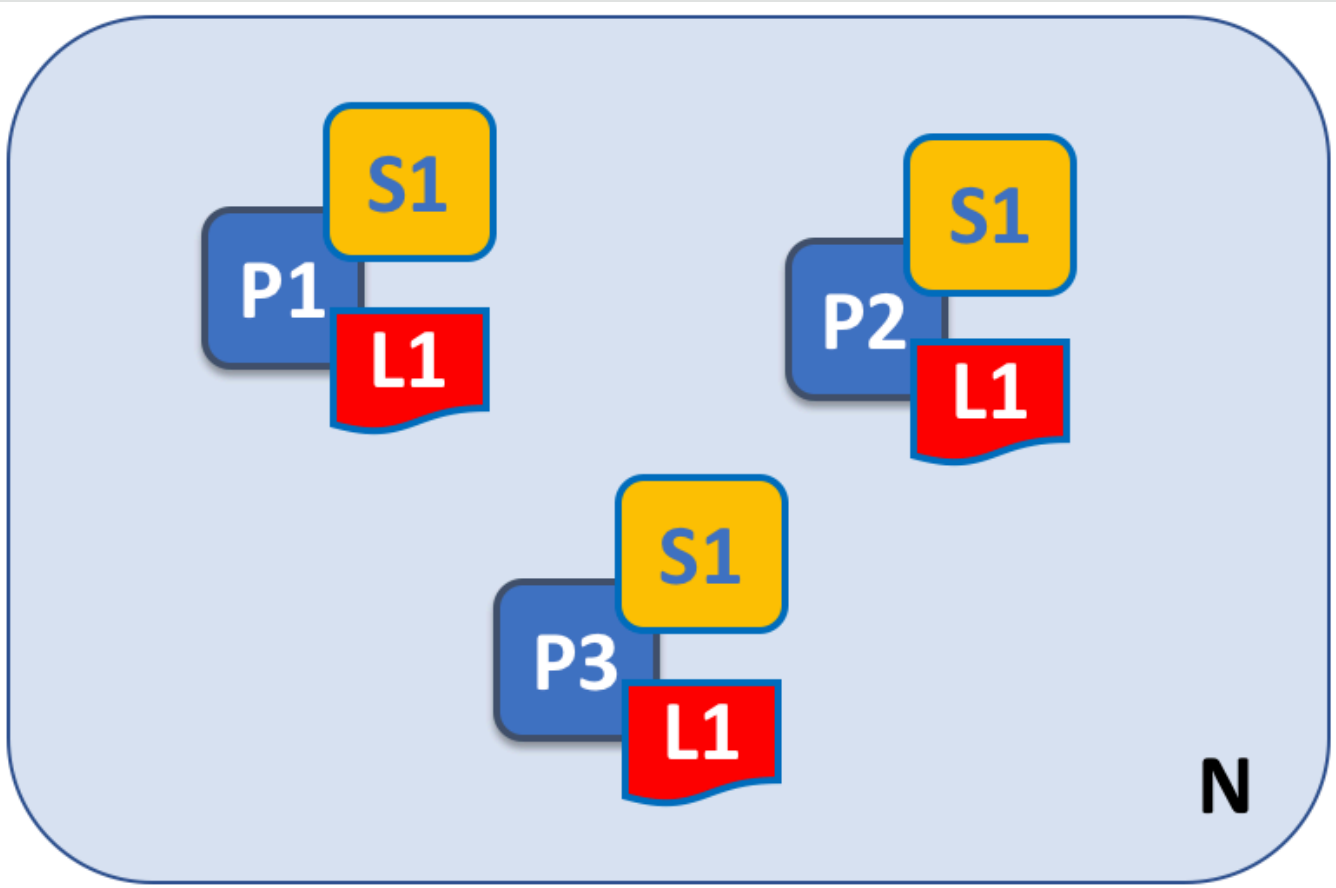
Hyperledger Fabric – Key Concepts

- Peers
- Channel
- Organization
- Ledgers
- Identity
- Membership

Peer의 종류

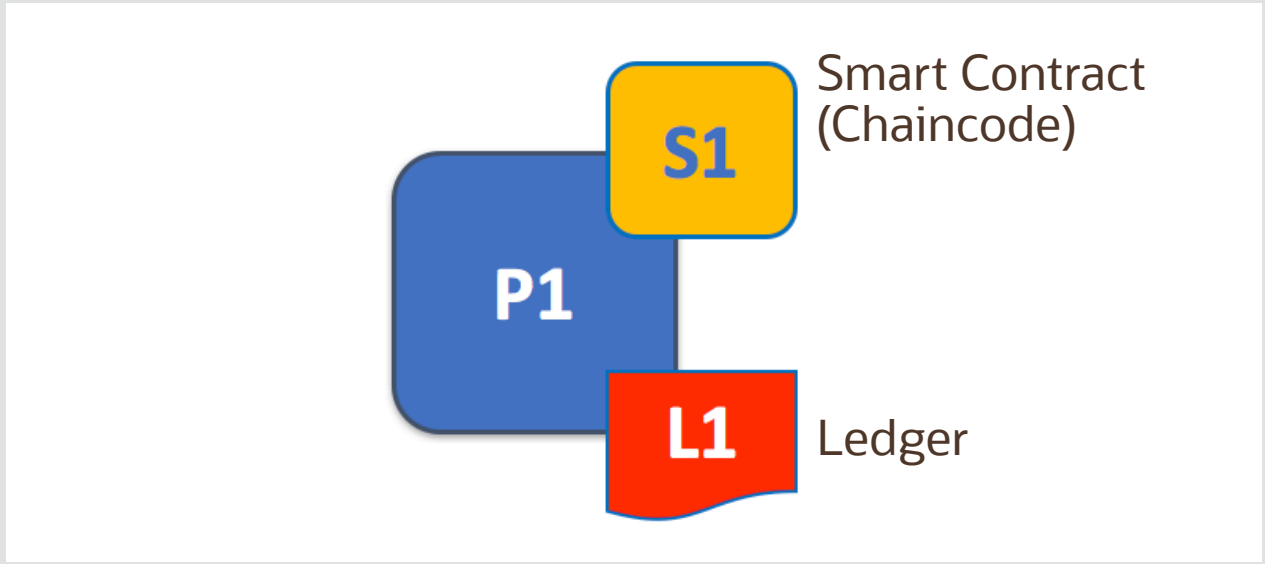
Peer	트랜잭션을 Commit, Ledger와 WorldState를 보관
Endorsing Peer	Endorse, Chaincode Execute
Ordering Peer	TX순서대로 정렬, 블록에 TX추가

Peer

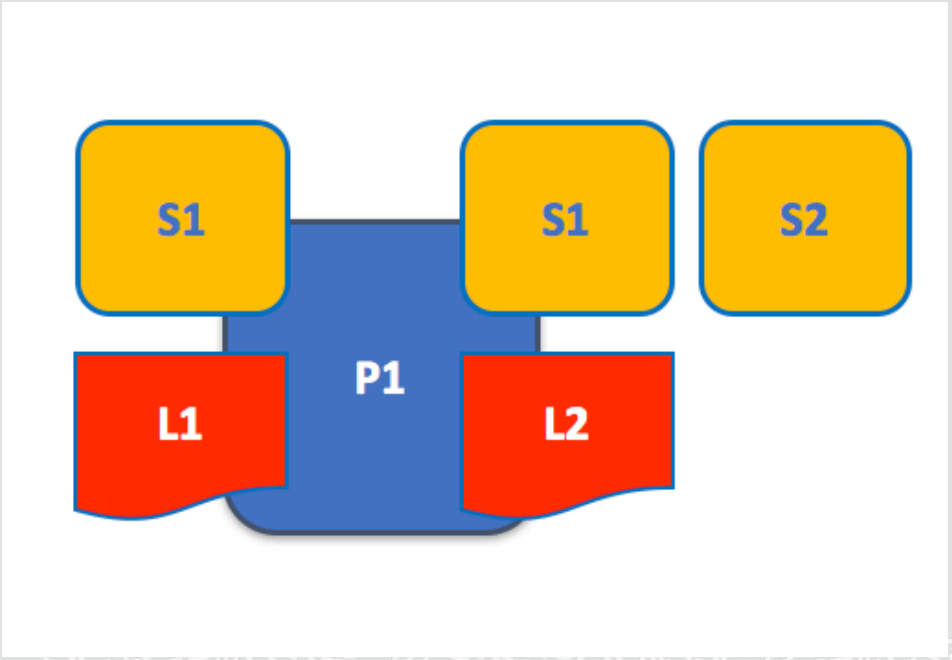


N	Blockchain network
P	Peer node
S	Smart contract (aka chaincode)
L	Ledger

Peers

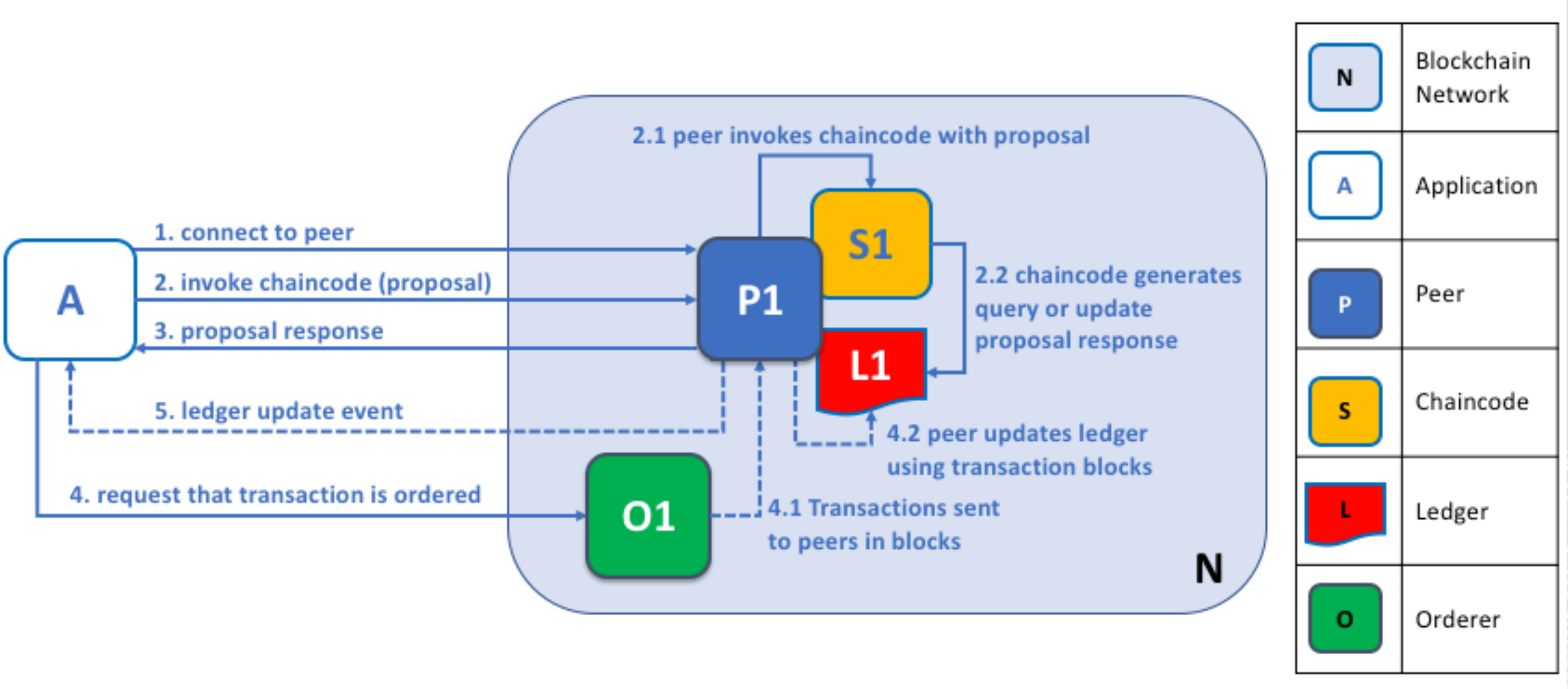


Peer

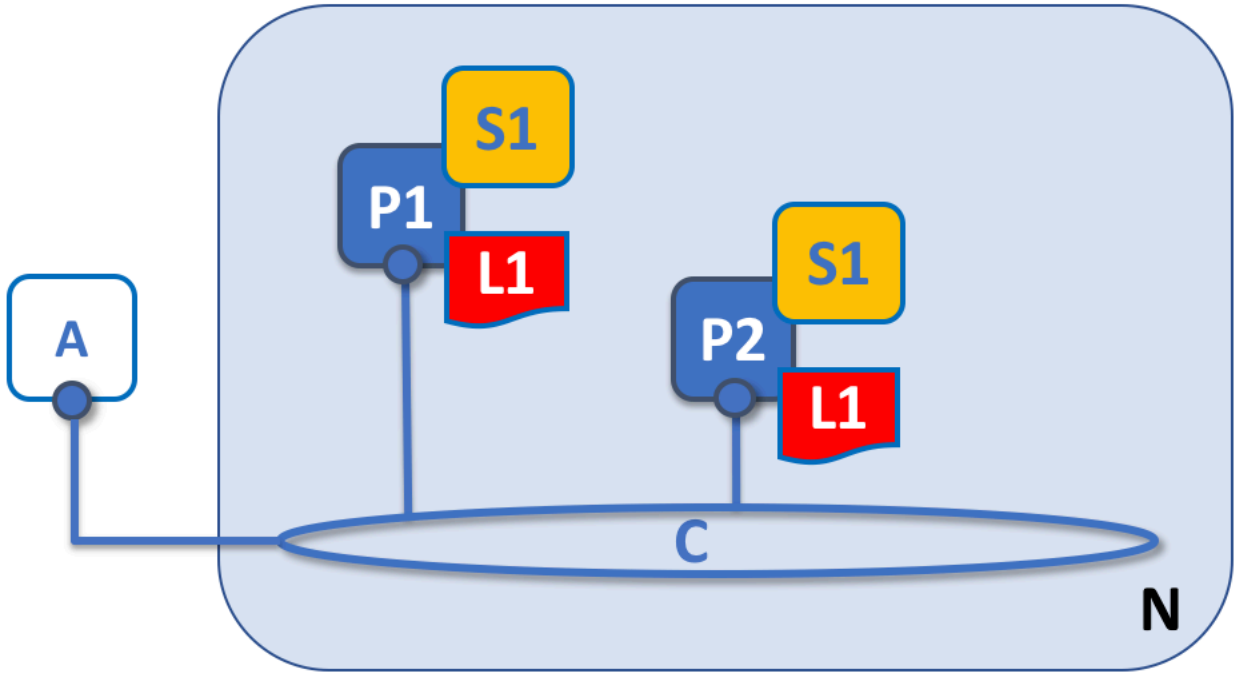


Multiple Ledgers

Peers

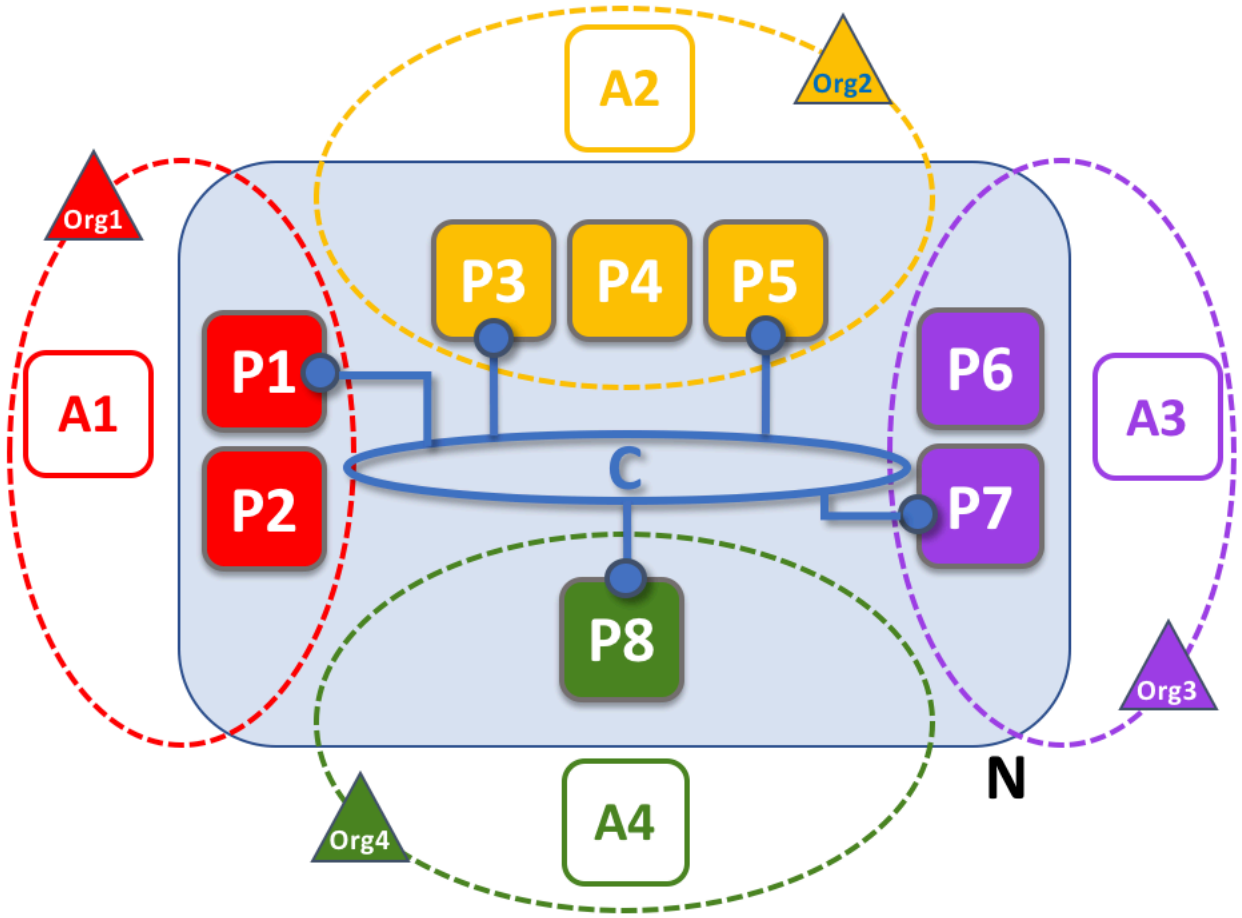

2.1 peer invokes chaincode with proposal
1. connect to peer
2. invoke chaincode (proposal)
3. proposal response
A
P1
S1
L1
2.2 chaincode generates query or update proposal response
5. ledger update event
4.2 peer updates ledger using transaction blocks
4. request that transaction is ordered
O1
4.1 Transactions sent to peers in blocks
N
N
Blockchain Network
A
Application
P
Peer
S
Chaincode
L
Ledger
O
Orderer

Channel



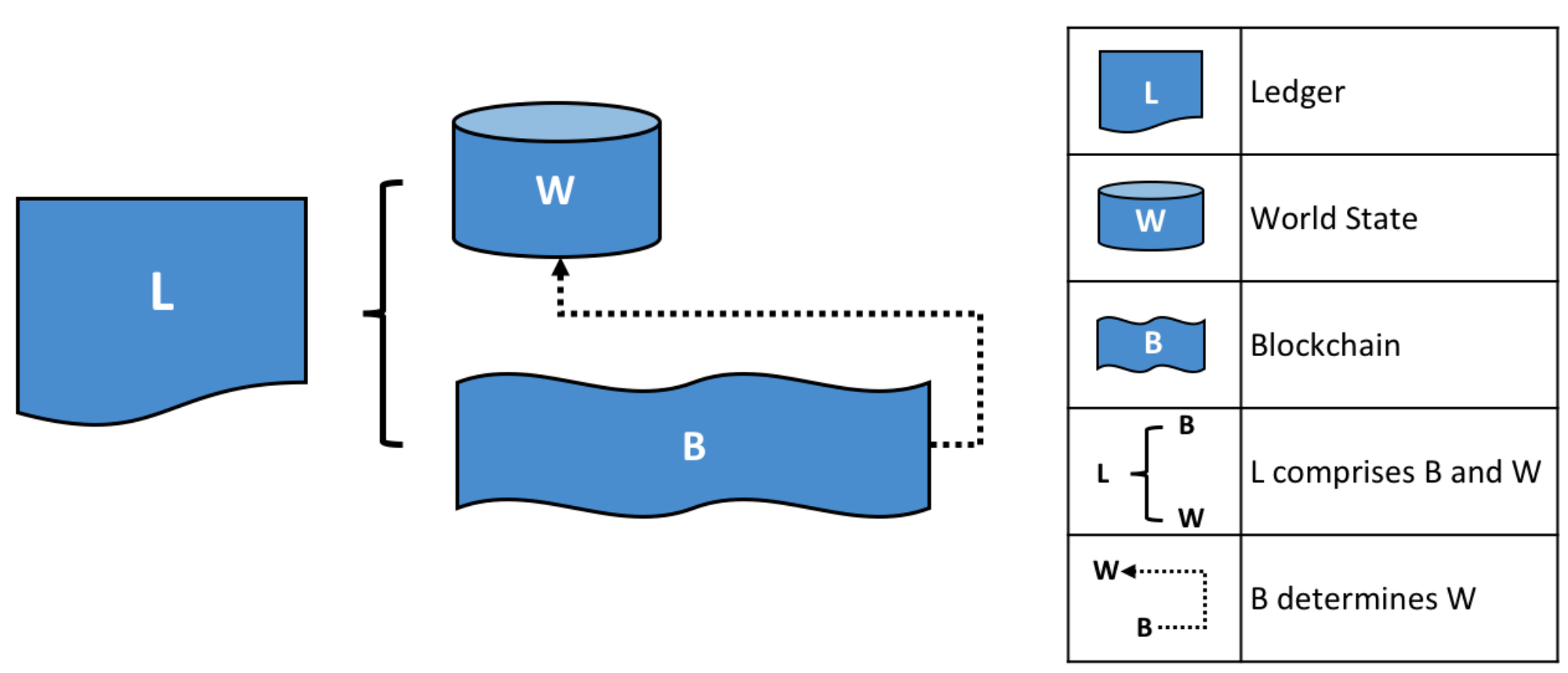
N	Blockchain Network	L	Ledger
C	Channel	A	Application
P	Peer	PA C	Principal PA (e.g. A, P1) communicates via channel C.
S	Chaincode		

Org(조직)

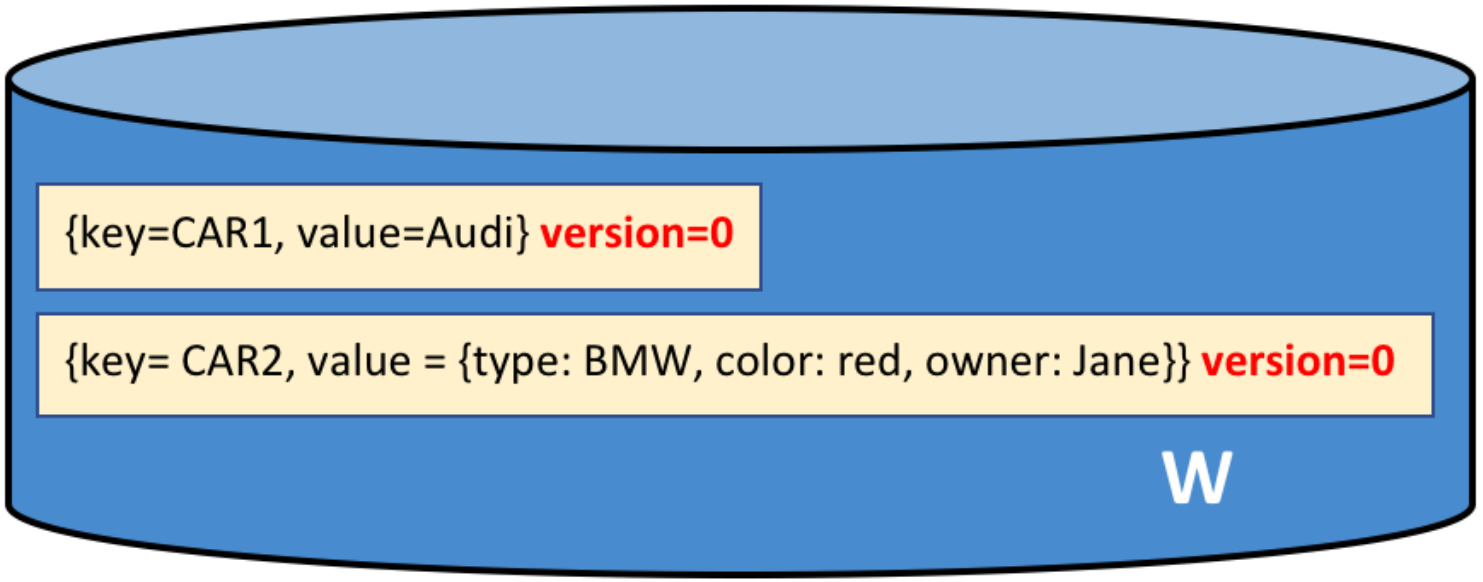


N	Blockchain Network	L	Ledger
C	Channel	A	Application
P	Peer	PA, C	Principal PA (e.g. A1, P5) communicates via channel C.
		Org	Organization
A1, R, P1, P2			Organization R owns application A1 and peers P1, P2.

Ledger


L
W
B
L
Ledger
W
World State
B
Blockchain
L
B
W
L comprises B and W
W
B
B determines W

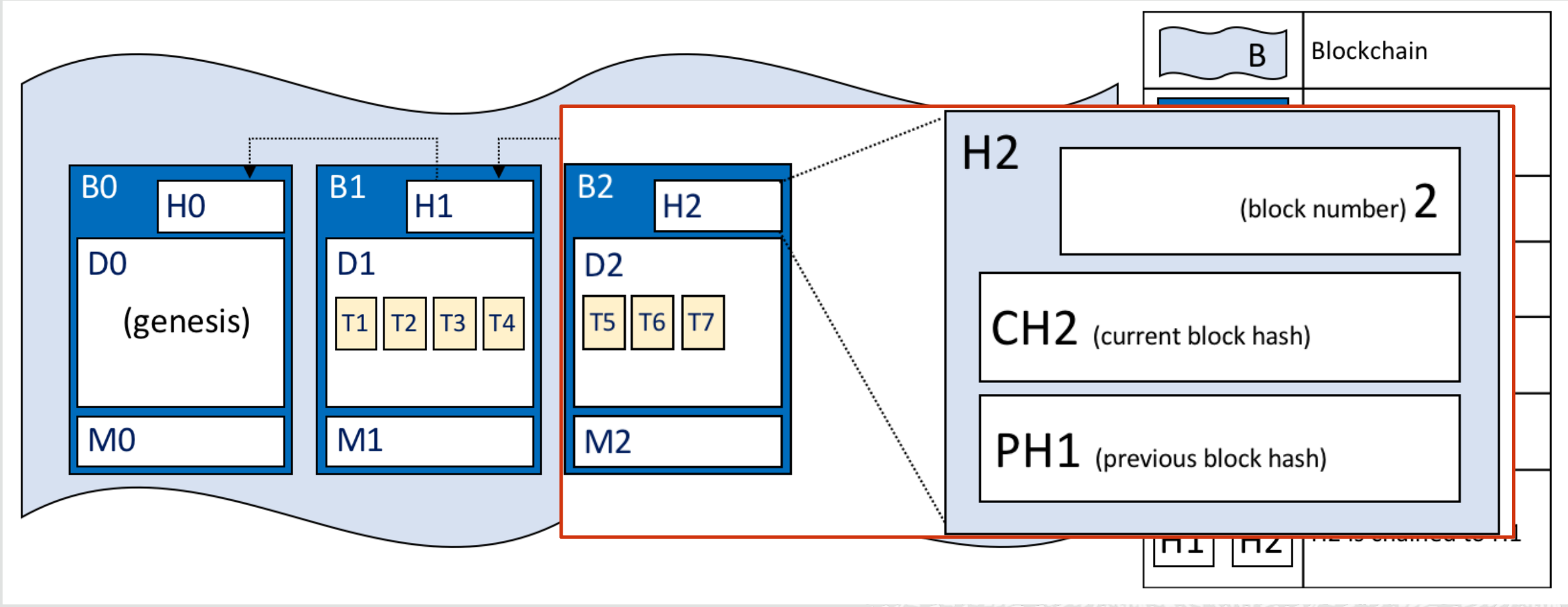
Ledger – world state



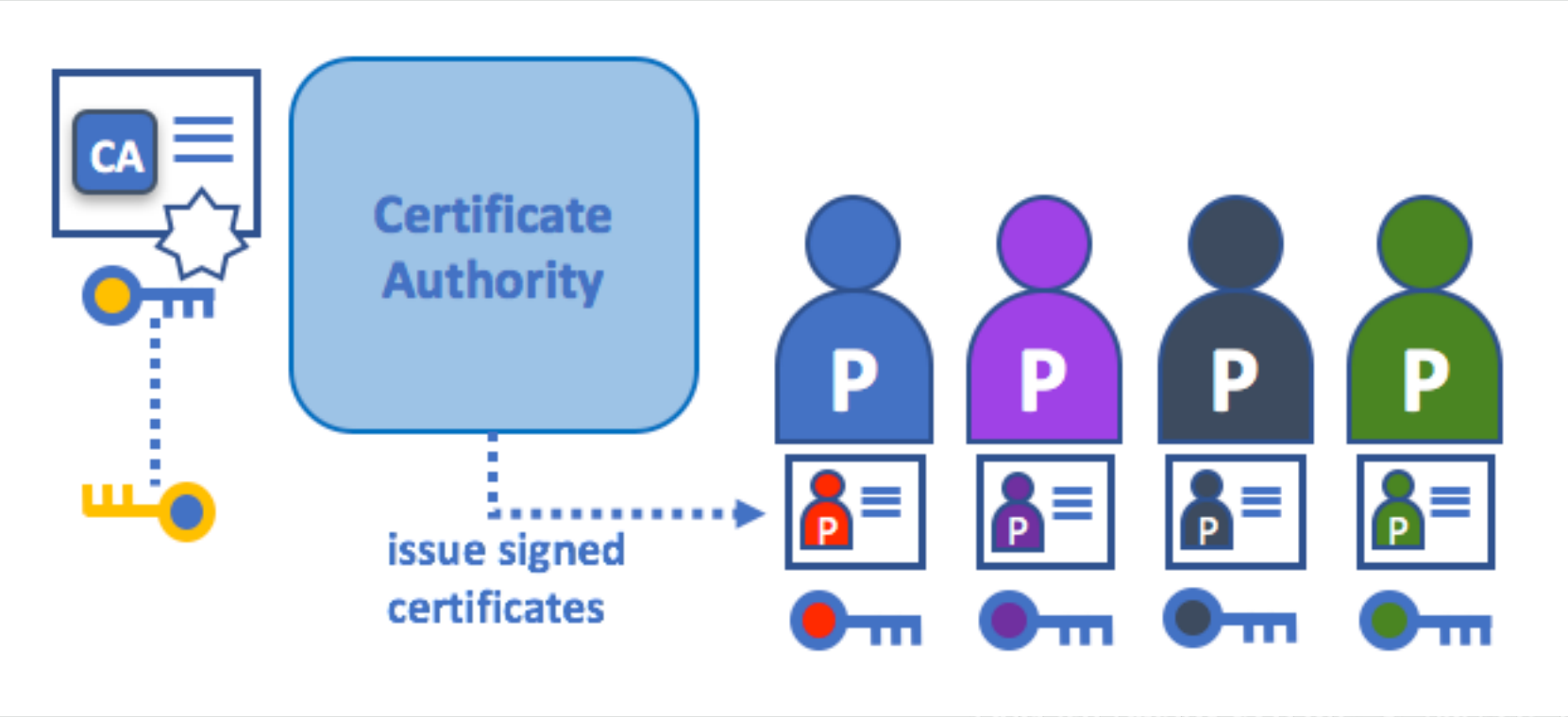
W	Ledger world state
{key=**K**, value = **V** } **version=0**	A ledger state with **key=K**. It contains a set of facts expressed as a simple value, **V**. The state is at version 0.
{key=**K**, value = {**KV**} } **version=0**	A ledger state with **key=K**. It contains a set of facts expressed as a set of key-value pairs **{KV}.** The state is at version 0.

LevelDB and CouchDB.

Ledger – blockchain

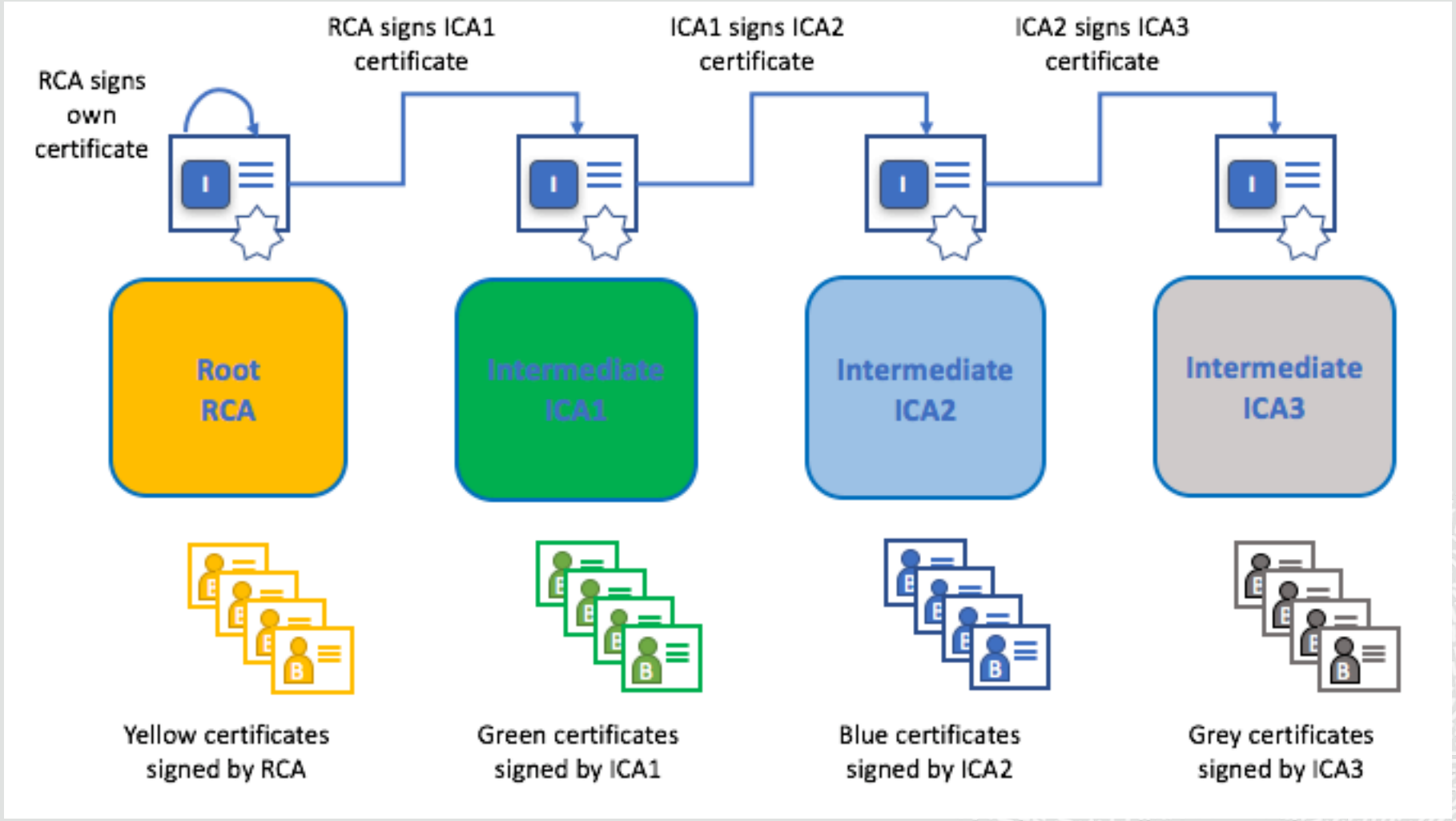


Identity – CA(Certificate Authorities)

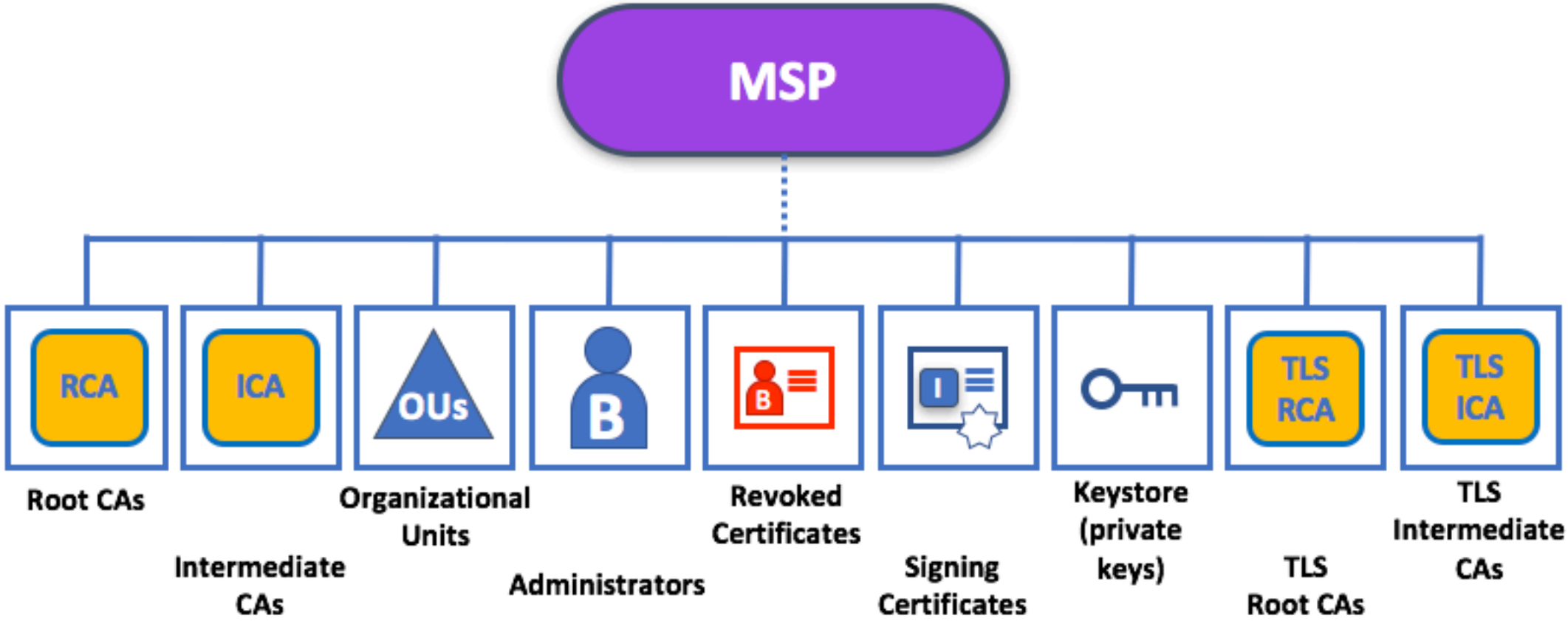


X.509

Identity – Root CAs, Intermediate CAs and Chains of Trust



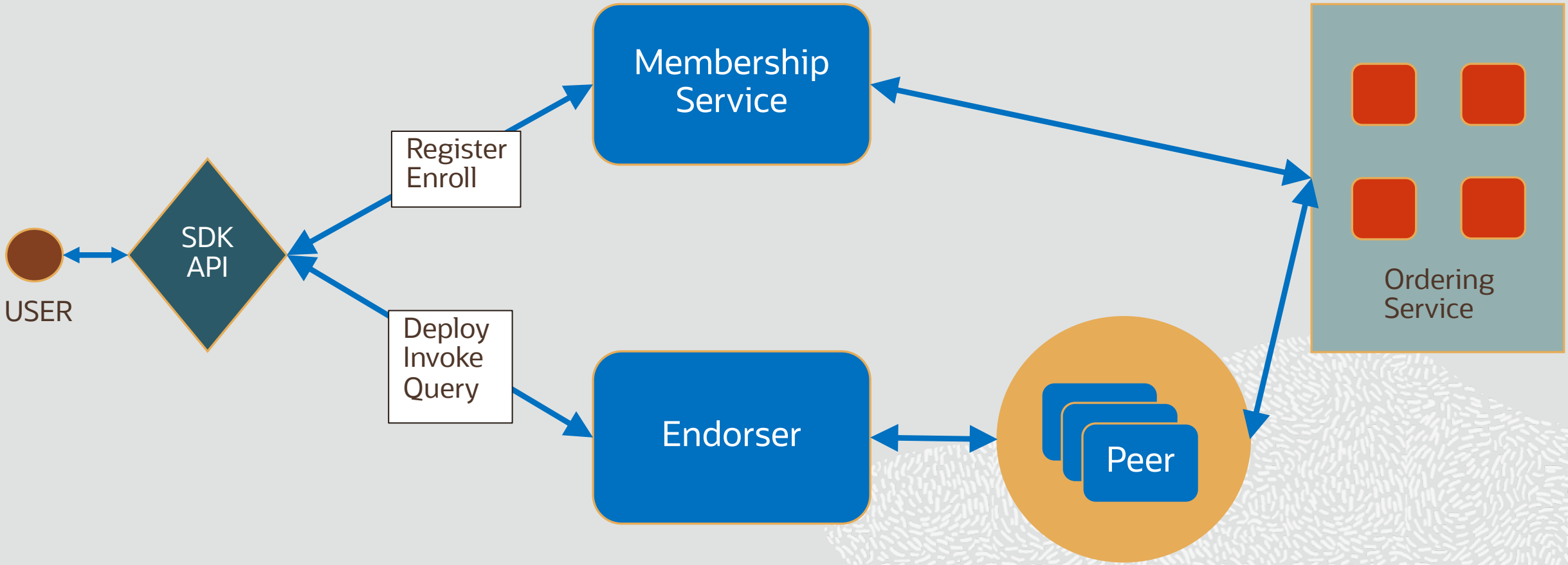
Membership - Membership Service Provider (MSP)



Architectures

Transaction Flows

Basic Architecture

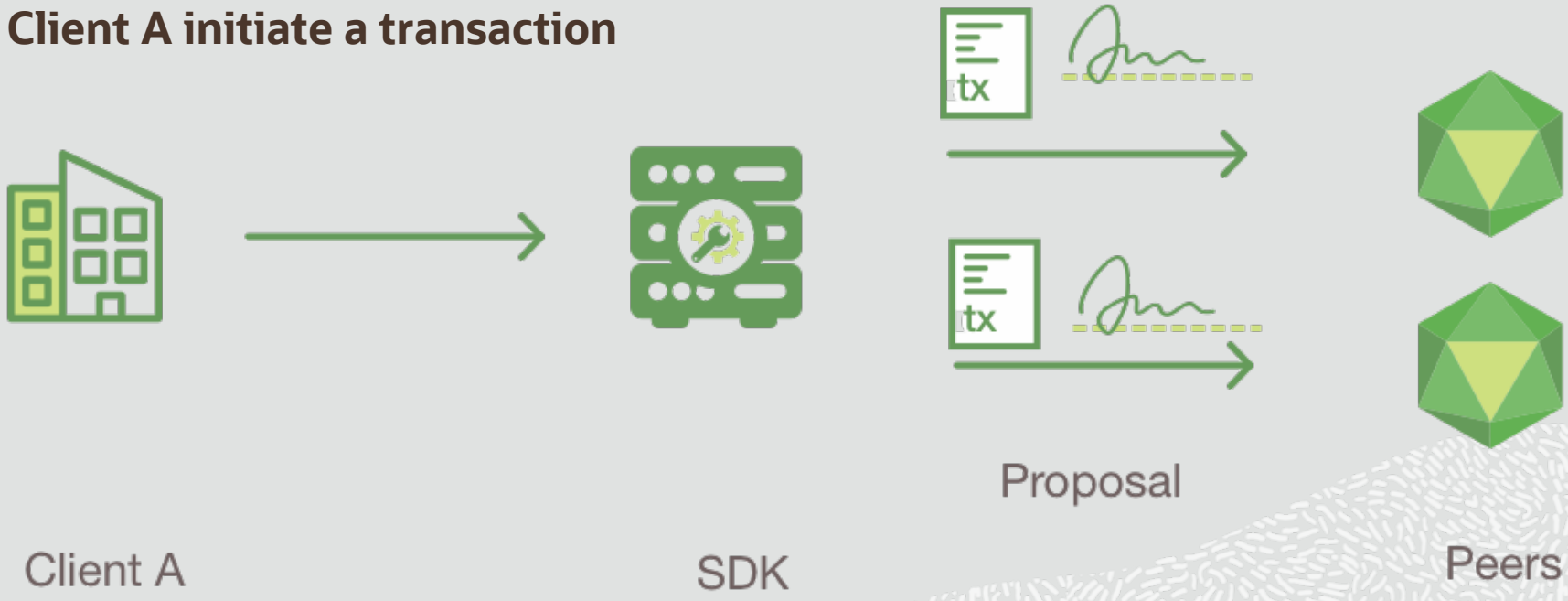


Transaction Flows

시나리오

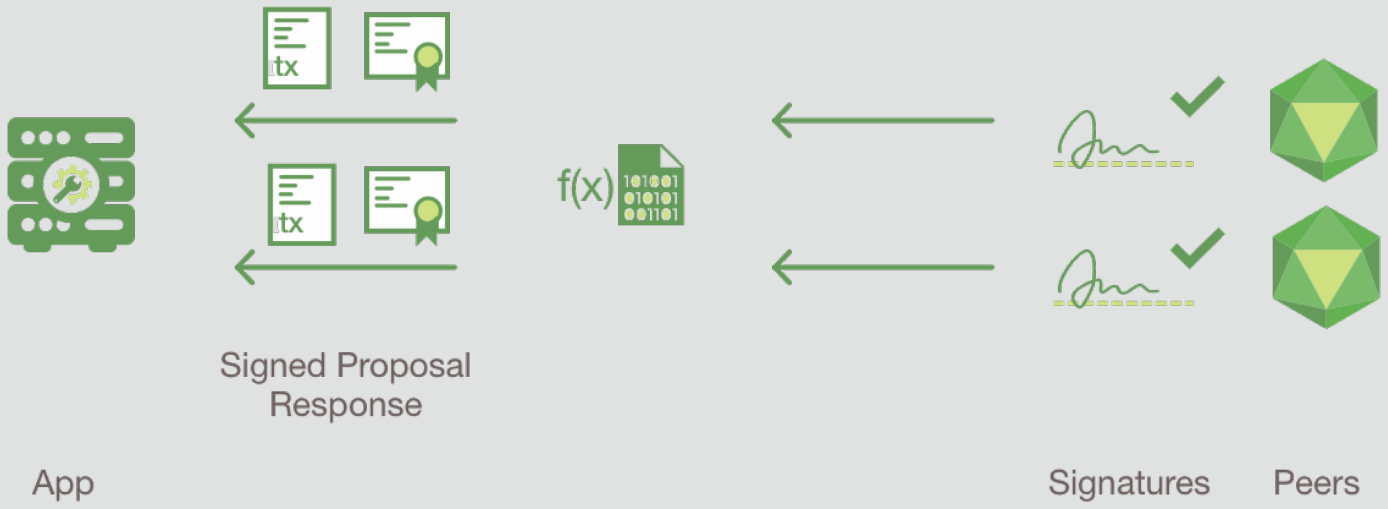
Transaction Flows

1. **Propose** - Client A initiate a transaction



Transaction Flows

2. Endorsing peers 서명을 검증하고, 트랜잭션을 실행



검증

(1) Transaction proposal 구조 확인,
(2) 이전에 실행되었는지 확인 (replay-attack protection),
(3) 서명이 유효한지 검증 (MSP)
(4) 권한 확인 (Client A)

실행

(1) Transaction 실행

응답

(1) 서명된 응답을 App(SDK)에 Return

Transaction Flows

3. Proposal responses are inspected

(1) endorsing peer signatures
(2) 모든 응답결과가 동일한지

Transaction Flows

4. Ordering - Client assembles endorsements into a transaction

<Transaction Message>

read/write set
Endorsing peer 서명
Channel ID

Ordering Service 는 검증을 하지 않고, 채널별로 트랜잭션들을 순서대로 정렬 → 블록생성.

Transaction Flows

5. Deliver – Transaction을 검증하고 Commit

채널에 있는 모든 Peers에게 블록 전달
Endorsement Policy에 맞는지 검증.
Transaction 수행시의 입력값과 동일한지 확인 후 valid여부 표시

Transaction Flows

6. Validate - Ledger updated

모든 피어에서는 블록을 각 채널의 체인에 추가한다.
Write set을 상태 database에 commit한다.

Hyperledger Fabric의 특징

Open source enterprise-grade permissioned distributed ledger technology (DLT) platform

Modularity : Pluggable(Orderer, membership service provider)

Permissioned Blockchains

Smart Contracts : Java, Go, Node.js

Privacy and Confidentiality

Pluggable Consensus : CFT (crash fault-tolerant) = Kafaka+Zookeeper

Performance and Scalability

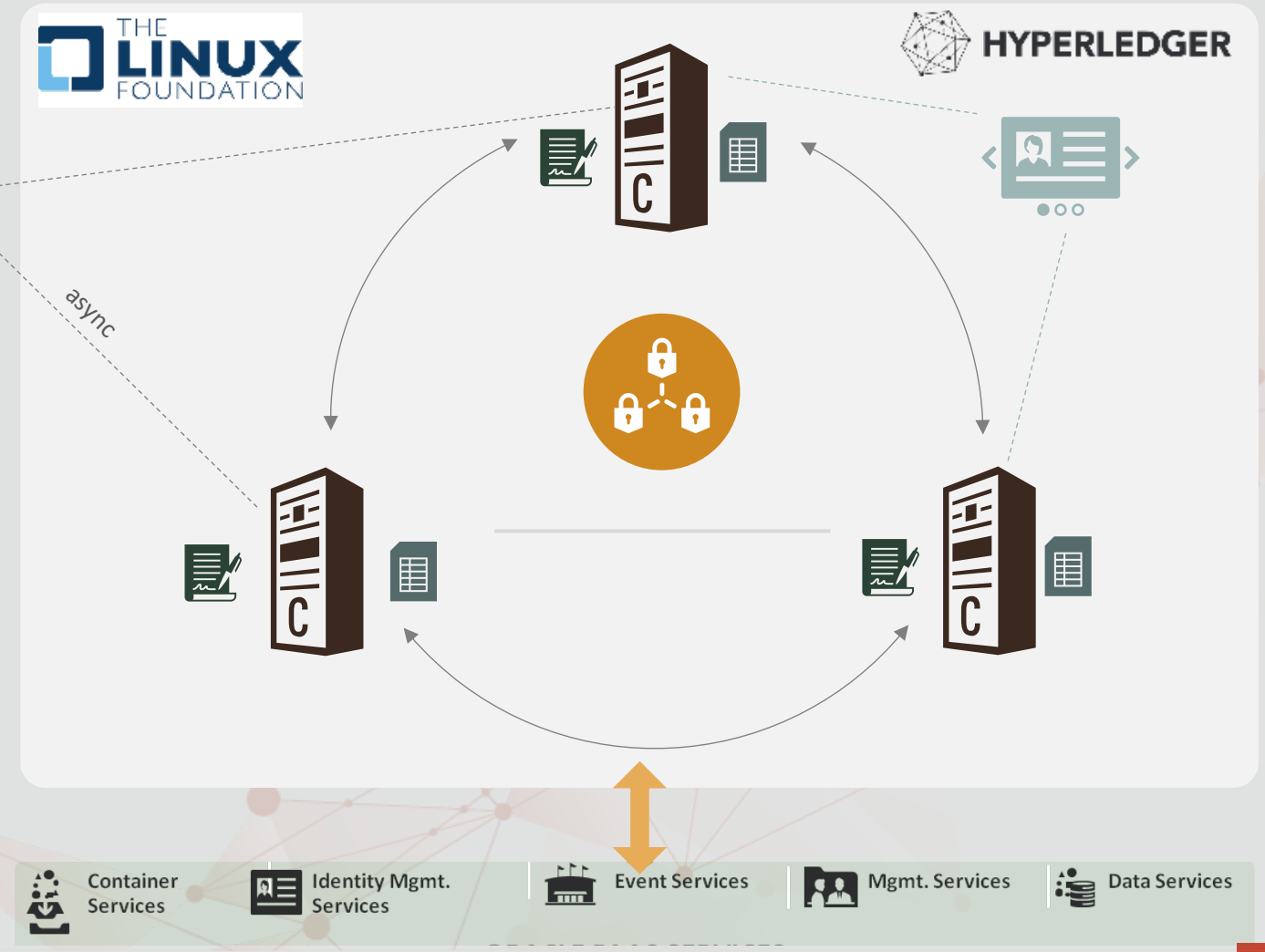
ORACLE

Oracle Blockchain Platform

기업용 블록체인에 필요한 사항

- Validating Nodes/ Peers
- Distributed Ledger (Single Version of Truth)
- Smart Contract (aka Chaincode)
- Ordering Service
- Membership Service

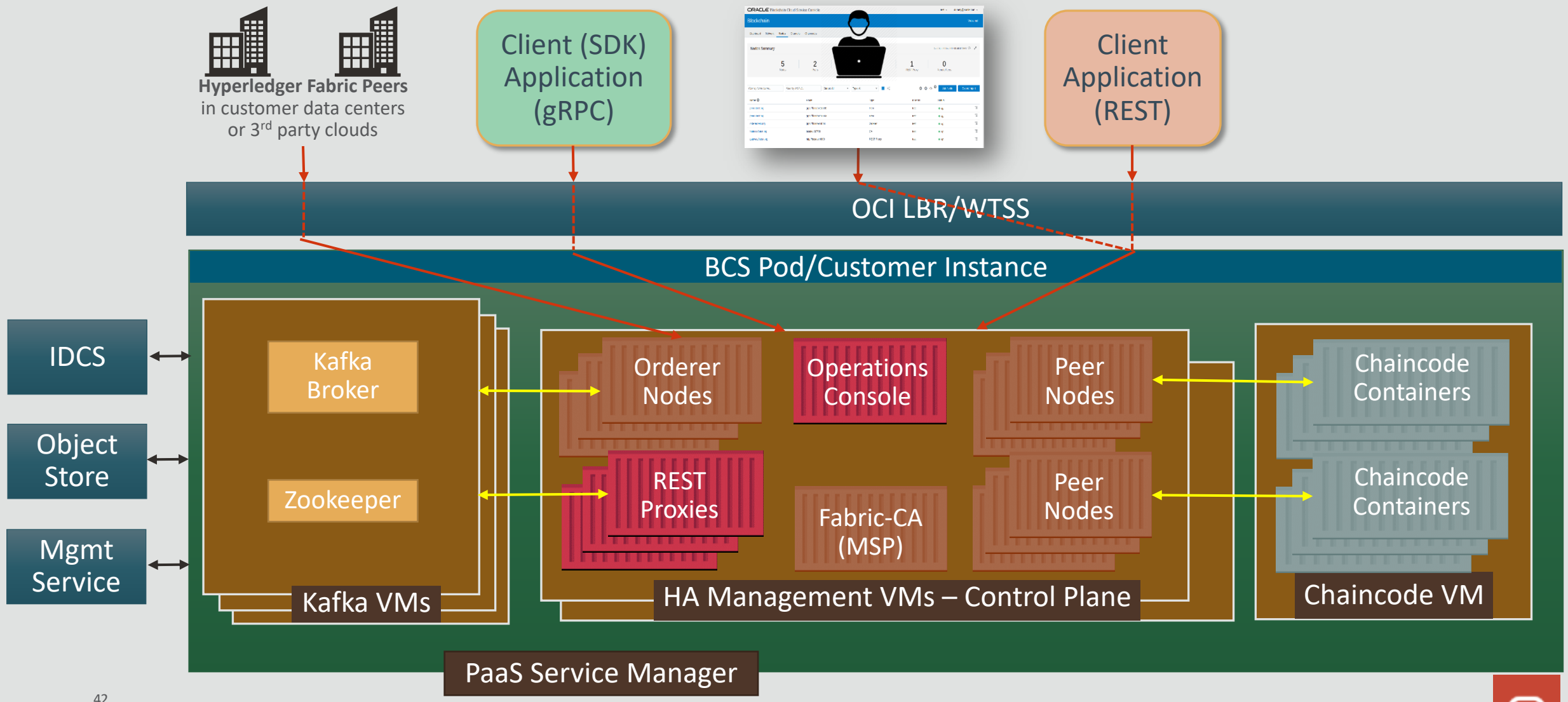
Applications (Client SDK)



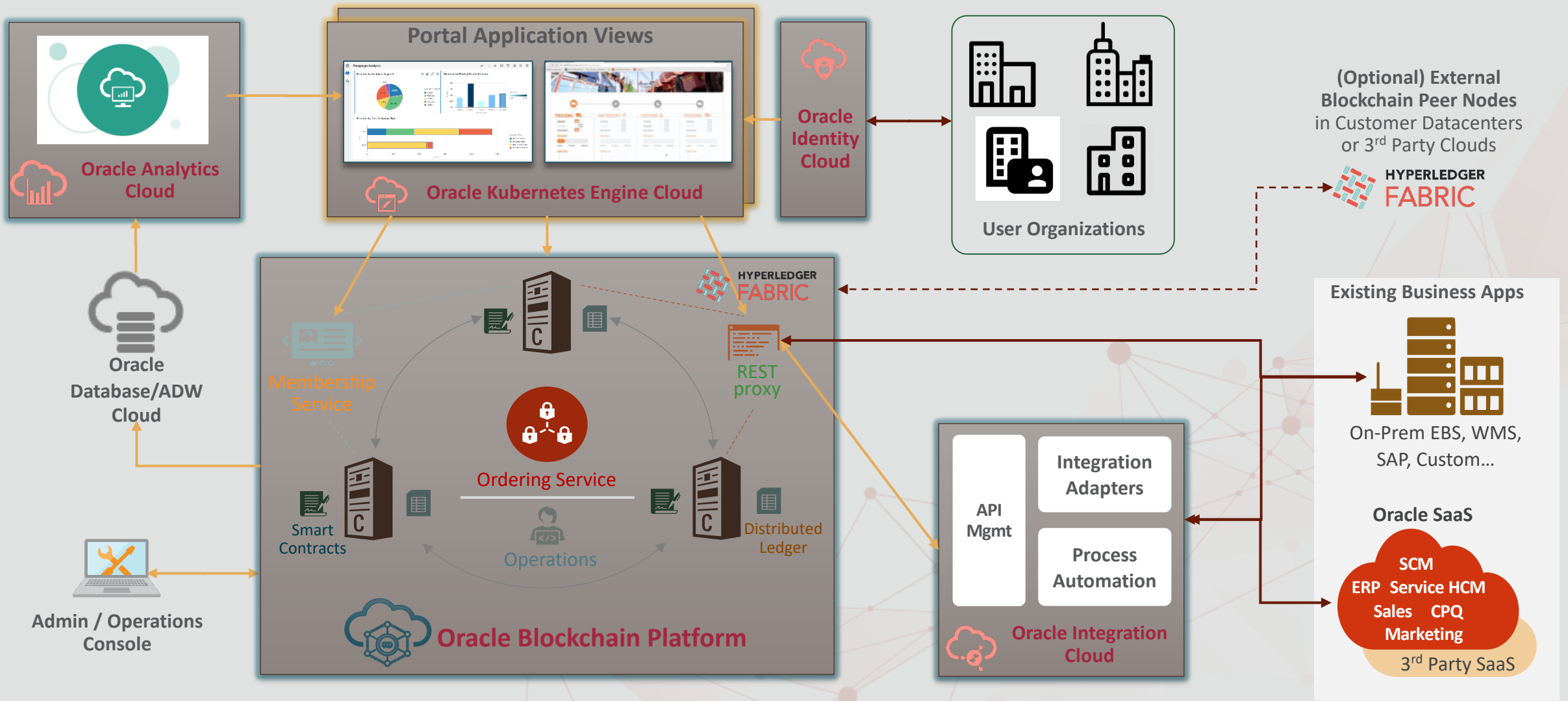
추가로 필요한 사항들

- Identity Management (for member enrollment)
- Fault Tolerant Event Hub (for ordering event mgmt.)
- Container Lifecycle Management (for system and chaincode containers)
- Configuration & Monitoring Tools
- Autonomous Recovery (when components fail)
- Managed Patching & Upgrades
- Elastic scaling on demand
- Multi-datacenter DR (with backup of ledger and config info)

OBP High-Level Service Architecture in OCI



Sample Solution Architecture



Provisioning 비교 : Native Hyperledger vs OBP

직접 Hyperledger Fabric 설치

OS 설치

cURL install

Docker / Docker Compose 설치

Go, Node.JS설치

Python설치

Curl로 Hyperledger 설치

Crypto-config.yml을 참조해서 Key file 생성

```
cryptogen generate --config=./crypto-config.yaml
```

Genesis block 생성

Crypto-config.yml을 참조해서 Key file 생성

```
configtxgen -profile OneOrgOrdererGenesis -
outputBlock ./config/genesis.block
```

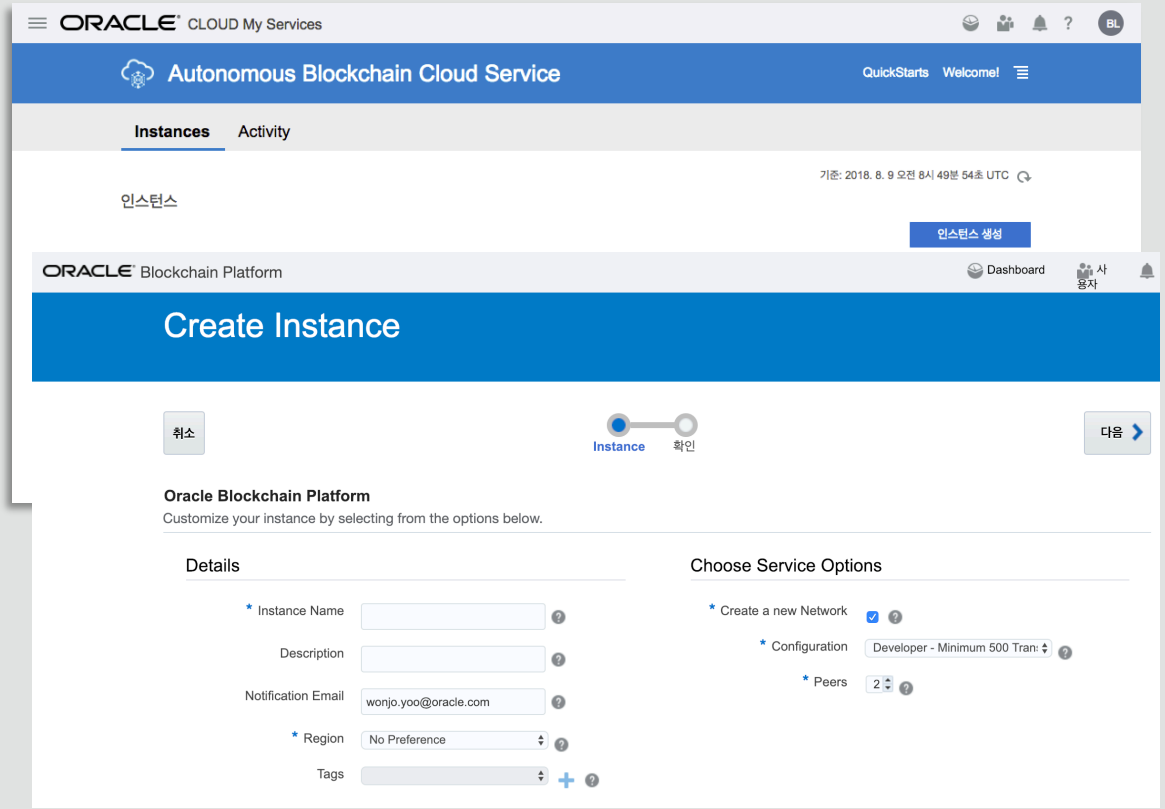
Create a Channel Configuration Transaction

```
../bin/configtxgen -profile TwoOrgsChannel -
outputAnchorPeersUpdate ./channel-
artifacts/Org1MSPanchors.tx -channelID $CHANNEL_NAME -
asOrg Org1MSP
```

Start the network

```
docker-compose -f docker-compose-cli.yaml up -d
```

Oracle Blockchain Platform



1 Step

Org 추가 및 Join 비교 : Native Hyperledger vs OBP

Hyperledger Fabric 에서 구성

docker-compose-neworg.yaml 생성

Generate the New Org Crypto Material

```
../../bin/cryptogen generate --config=./org3-crypto.yaml
cd ../ && cp -r crypto-config/ordererOrganizations org3-artifacts/crypto-config/
```

Prepare the CLI Environment

Fetch the Configuration

```
peer channel fetch config config_block.pb -o orderer.example.com:7050 -c $CHANNEL_NAME --tls --cafile $ORDERER_CA
```

Convert the Configuration to JSON and Trim It Down

```
configtxlator proto_decode --input config_block.pb --type common.Block | jq .data.data[0].payload.data.config > config.json
```

Add the Org3 Crypto Material

```
jq -s '.[0] * {"channel_group":{"groups":{"Application":{"groups": {"Org3MSP":.[1]}}}}}' config.json ./channel-artifacts/org3.json > modified_config.json
configtxlator proto_encode --input config.json --type common.Config --output config.pb
5 more steps …
```

Sign and Submit the Config Update

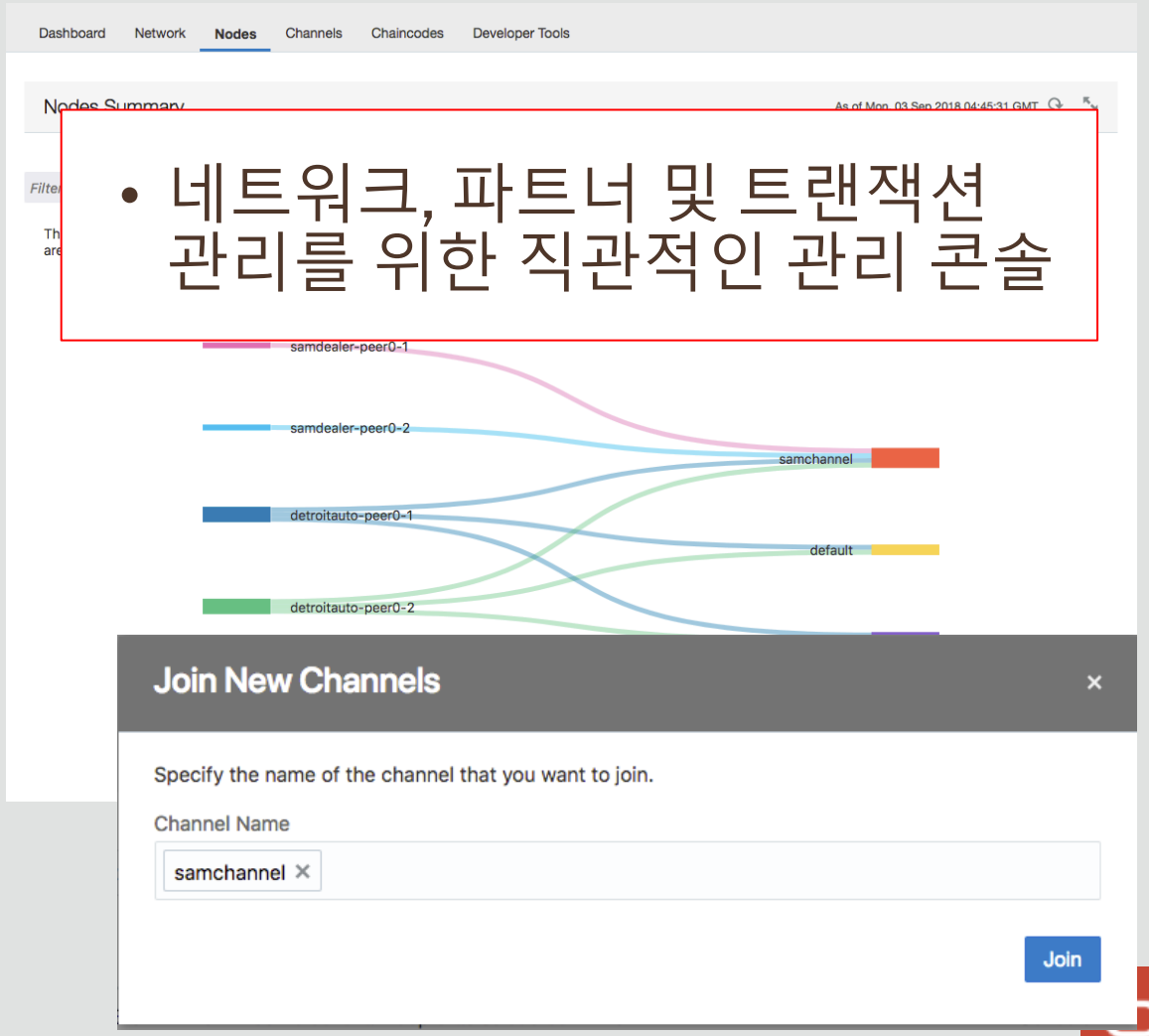
```
peer channel signconfigtx -f org3_update_in_envelope.pb
peer channel update -f org3_update_in_envelope.pb -c $CHANNEL_NAME -o orderer.example.com:7050 --tls --cafile $ORDERER_CA
```

Configuring Leader Election

Join Org3 to the Channel

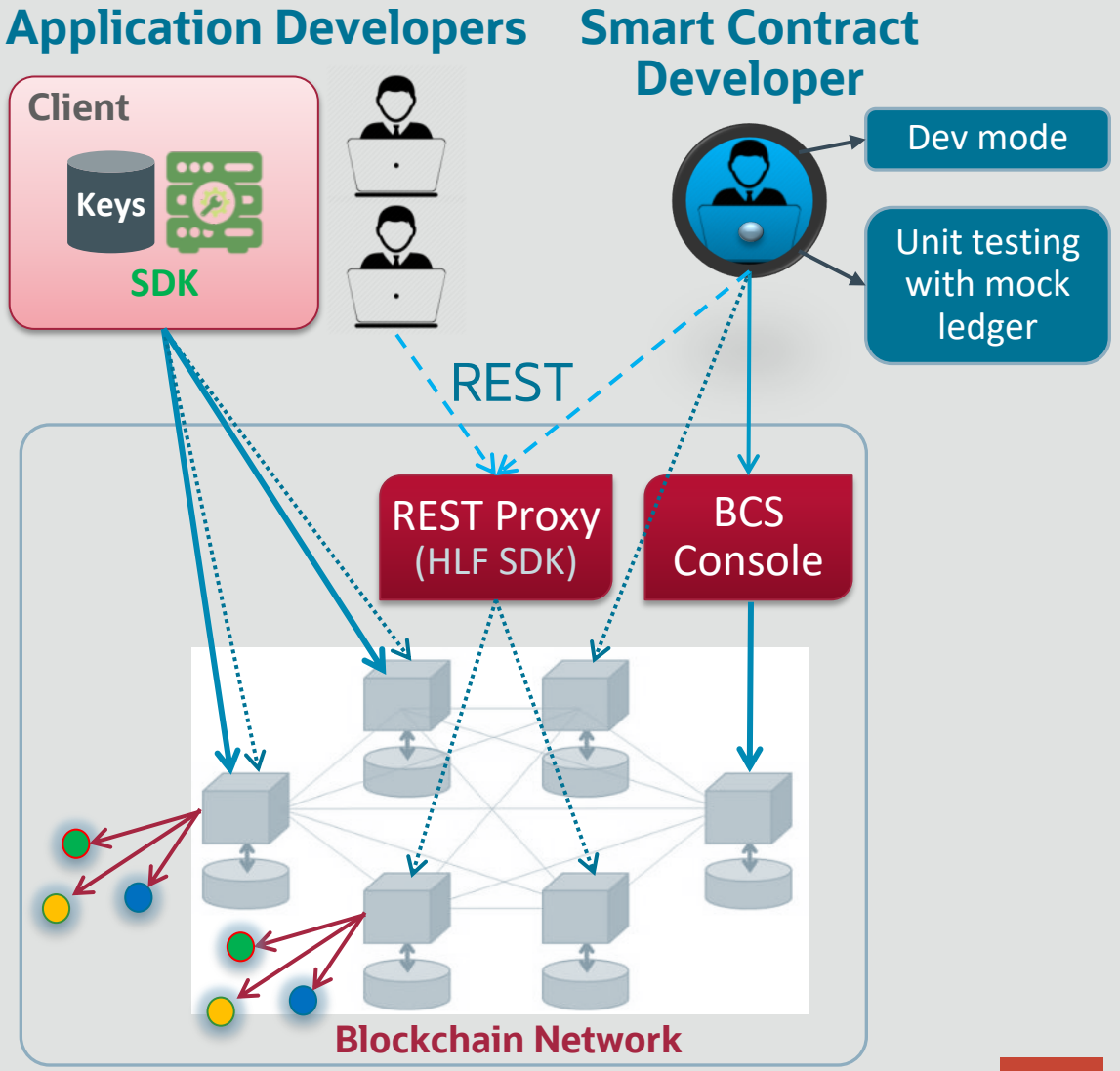
```
docker-compose -f docker-compose-org3.yaml up -d
peer channel fetch 0 mychannel.block -o orderer.example.com:7050 -c $CHANNEL_NAME --tls --cafile $ORDERER_CA
peer channel join -b mychannel.block
```

Oracle Blockchain Platform



Smart Contract, Application 개발

- **Smart Contract** 개발자는 블록 체인에서 비즈니스 로직을 만들고 테스트
  - 콘솔 또는 SDK를 통해 배포
  - 웹 서비스 / REST API 또는 SDK를 통한 테스트
- **Application** 개발자는 배포 된 Smart Contract를 통해 트랜잭션을 실행하는 최종 사용자 응용 프로그램을 개발
  - Client SDK (Java, Go, node.js)
  - Web Services/REST APIs
- **개발자 생산성 향상**
  - Rich queries on history DB
  - Developer Cloud를 CI / CD 용 DevOps 도구로 이용



Chaincode 배포 비교 : Native Hyperledger vs OBP

Hyperledger Fabric 에서 배포

Chaincode 목록 확인

peer chaincode list --installed

peer chaincode list --instantiated -C org1mspc

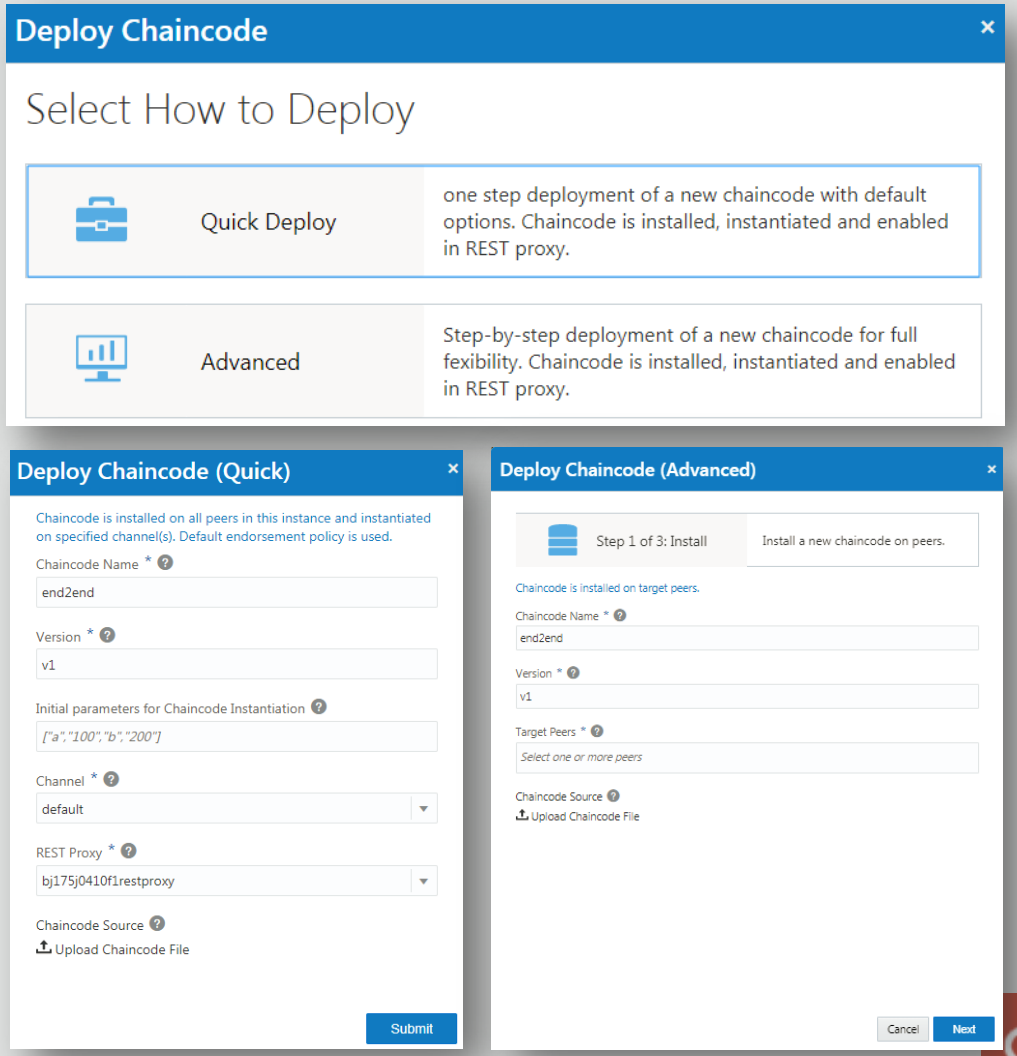
Chaincode install

peer chaincode install -n mycc -v 1.0 -l “golang" -p ${CC_SRC_PATH}

Chaincode instantiate

peer chaincode instantiate -o ${orderer_addr}:${orderer_port} --tls --cafile orderer.pem -C mychannel -n mycc -l golang -v 1.0 -c '{"Args":["init","a","100","b","200"]}' -P <policy_string> --logging-level debug

Oracle Blockchain Platform



개발 편의성 – SQL기반 Query 사용

Couch DB Query Syntax : OBCS에서 호환됨

```
func (t *CarChaincode) queryUserByNamePhone(stub shim.ChaincodeStubInterface, args []string) peer.Response {
    queryString := fmt.Sprintf("{\"selector\":{\"UserName\":\"%s\",\"PhoneNo\":\"%s\"}}", name, phone)
```

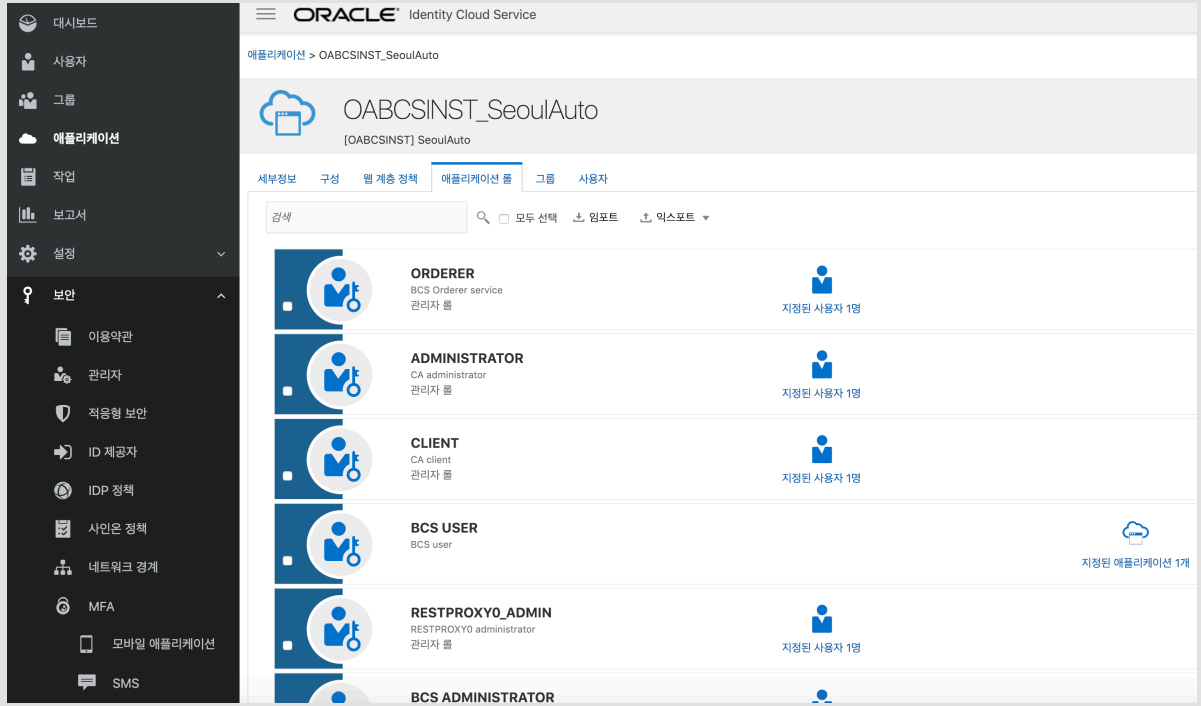
Berkeley DB Query Syntax : SQL 형태로 되어 있어 사용이 편리함

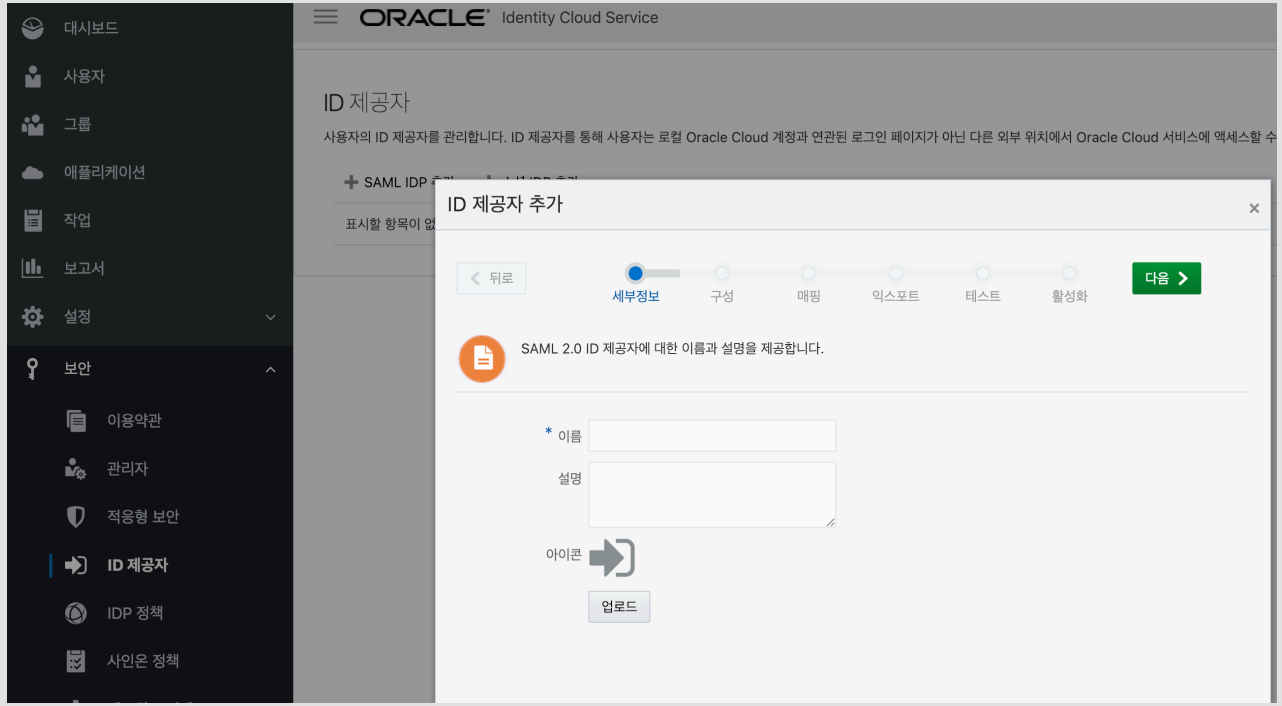
```
queryString := fmt.Sprintf("SELECT valueJson FROM <STATE> WHERE json_extract(valueJson, '$.docType', '$.UserName', '$.PhoneNo')
```

```
func (t *HanaChaincode) queryUserByIdLike(stub shim.ChaincodeStubInterface, args []string) peer.Response {
    queryString := fmt.Sprintf("SELECT valueJson FROM <STATE> WHERE key like '%s%%'", id)
```

계정 관리 시스템 연계

- Oracle 클라우드의 계정관리 시스템과 Hyperledger의 CA와 계정 연계
- Third Party 계정 Provider와의 연동 - SAML지원, Microsoft Active Directory 연동, SSO
- 사용자별 ACL 관리





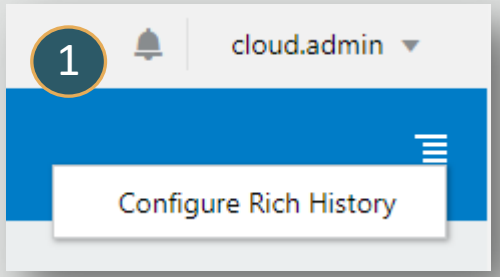
Rich History DB for Analytics Integration

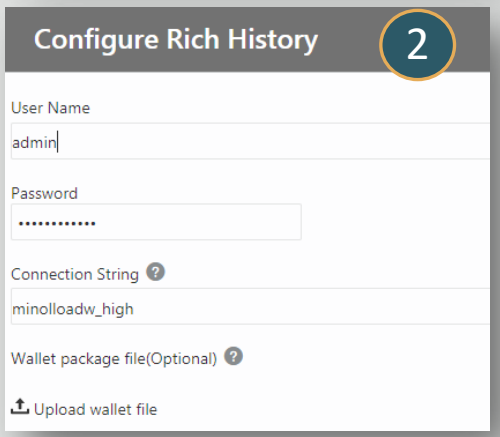
Parallel with regular history DB updates, we asynchronously up date Oracle ADW/DBaaS for every transaction commit

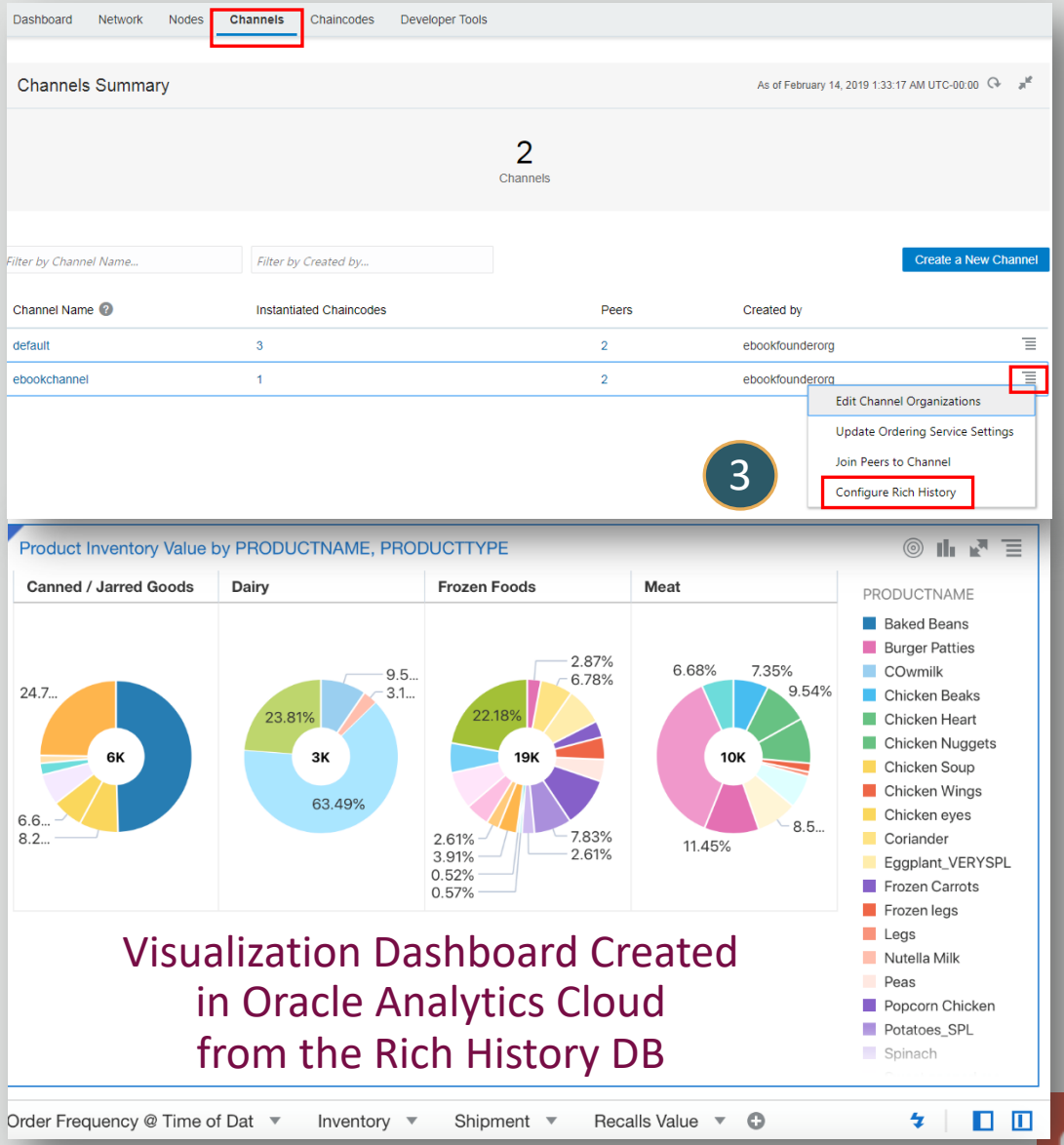
DB maintains rich data model using Oracle JSON support (can be unpacked in OAC project)

Accessible for Analytics / BI / D WH reporting, interactive visuali zation dashboards, etc.

Can be used for transaction con firmations and high volume rea d-only access when async dela y is not critical

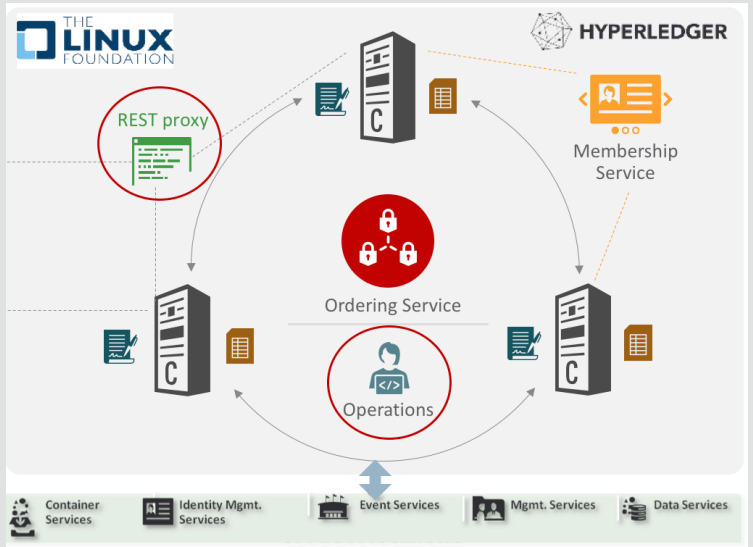






Visualization Dashboard Created in Oracle Analytics Cloud from the Rich History DB

Easily Integrate and Develop Applications



Apps Apps

- Oracle SaaS SCM 및 ERP 애플리케이션 확장을 위한 사용 준비된 SDKs
- Oracle 또는 타사 애플리케이션에서 블록체인 트랜잭션을 트리거하는 간편한 REST API
- Oracle PaaS 서비스에 대한 블록체인 애플리케이션을 신속하게 개발

ORACLE

Oracle Apps Leveraging Blockchain Platform

API-Based Integrations

ORACLE NETSUITE

40,000+ Global Organizations

ORACLE FLEXCUBE

319 Banks in more than 115 countries

ORACLE OPEN BANKING API PLATFORM

Oracle's PaaS/SaaS curated Fintech AP

ORACLE RETAIL

Brand Compliance in Grocery Supply Chain

ORACLE HEALTH SCIENCES

Clinical trial data collection & reporting

ORACLE CONSTRUCTION AND ENGINEERING

SaaS on OBP

ORACLE CLOUD SCM

- Intelligent Track and Trace
- Lot Lineage and Provenance
- Intelligent Cold Chain
- Warranty and Usage Tracking

OABCS Smart Contracts, Distributed Ledger

Shipment Notifications, Bill of Lading, Manufacturing work orders	Purchase order, Sales order	Service records, Warranty information	Authentication, Immutability, Trust, Consensus	Cargo conditions, Predictive Insights

SOURCES OF INPUT

ORACLE SCM CLOUD | ORACLE ERP CLOUD | ORACLE CUSTOMER EXPERIENCE CLOUD | ORACLE BLOCKCHAIN CLOUD | ORACLE IOT CLOUD

Customer Driven

ORACLE E-BUSINESS SUITE

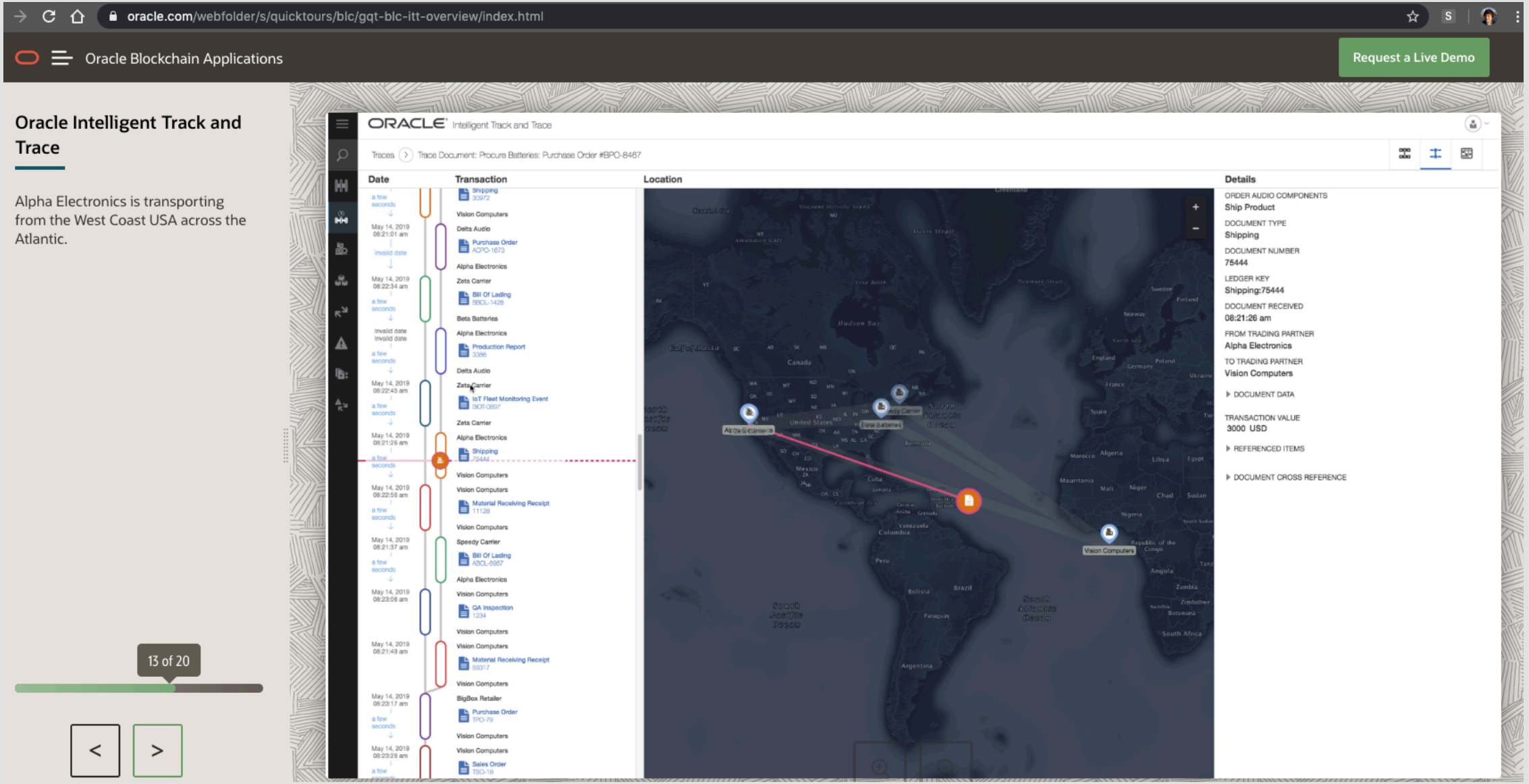
ORACLE FIELD SERVICE CLOUD

ORACLE CUSTOMER EXPERIENCE CLOUD

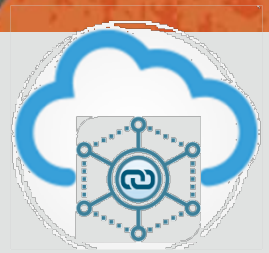
ORACLE SIEBEL DMS

*coming soon

Oracle Intelligent Track and Trace



QuickStart Templates

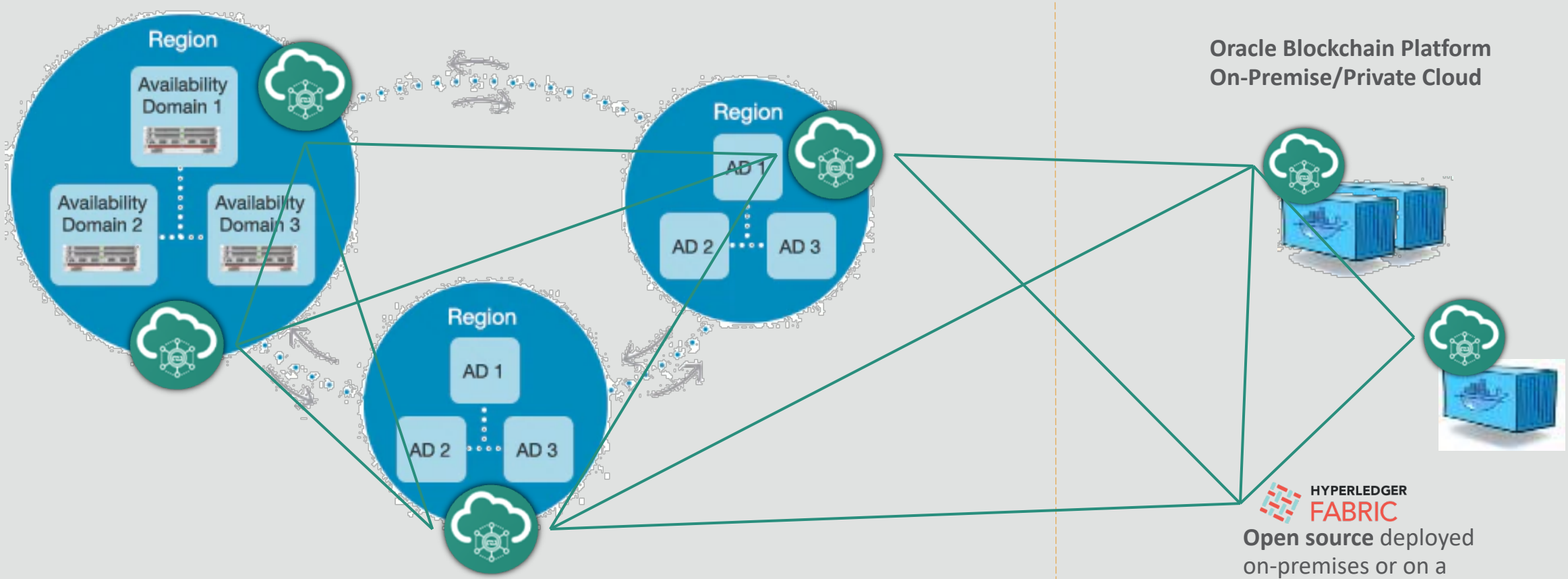


	Developer	Enterprise X1	Enterprise X4
BCS Control VMs	1	**2**	2
Chaincode VMs	1	1	1
VM Shapes	2.1 (1 OCPU)	2.1 (1 OCPU)	**2.4 (4 OCPU)**
Kafka	1 node in control VM	**3 node cluster in 3 VMs**	3 node cluster in 3 VMs
Max Peer Nodes	4	**14**	14
Orderer Nodes	1	**2**	2
Storage	1TB Block + 1TB OSS	2TB Block + 1 TB OSS	2TB Block + 1 TB OSS
Minimum Consumption	1 unit (500 txn/hr)	**2 units (1000 txn/hr)**	**8 units (4000 txn/hr)**
Minimum Monthly Flex	$372	$744	$2,880

Flexible, Global Blockchain Network Topologies

Oracle Cloud Datacenters

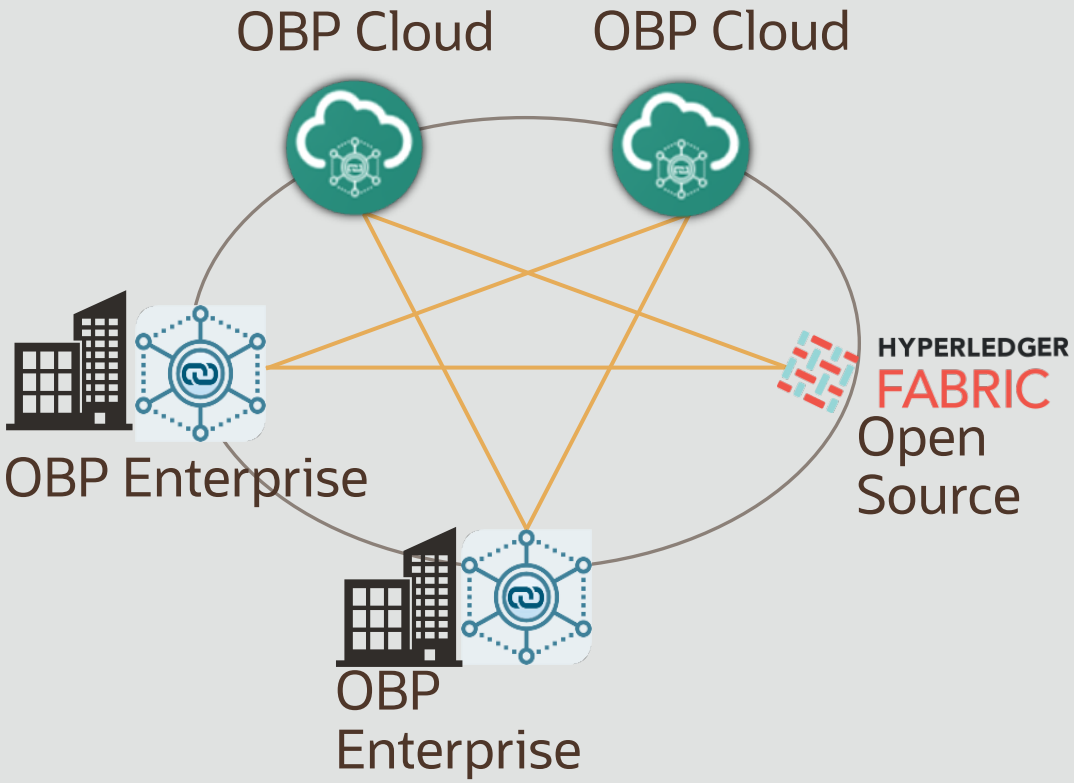
Customer Datacenters



*coming soon

Oracle Blockchain Platform Enterprise Edition

- Deploy Oracle Blockchain in your data center
  - Choice of virtualization platforms
- Create Blockchain network with a few clicks
  - Fully pre-assembled
- Feature parity with Blockchain Cloud
  - Same APIs & portability of applications
- Support for hybrid networks across
  - Oracle Cloud & 3rd party Blockchains



Oracle Blockchain Platform Enterprise Edition

1. Blockchain Platform Manager : quick **provisioning**, **patching** of your blockchain instances
All components required to create and run a Hyperledger Fabric-based blockchain network

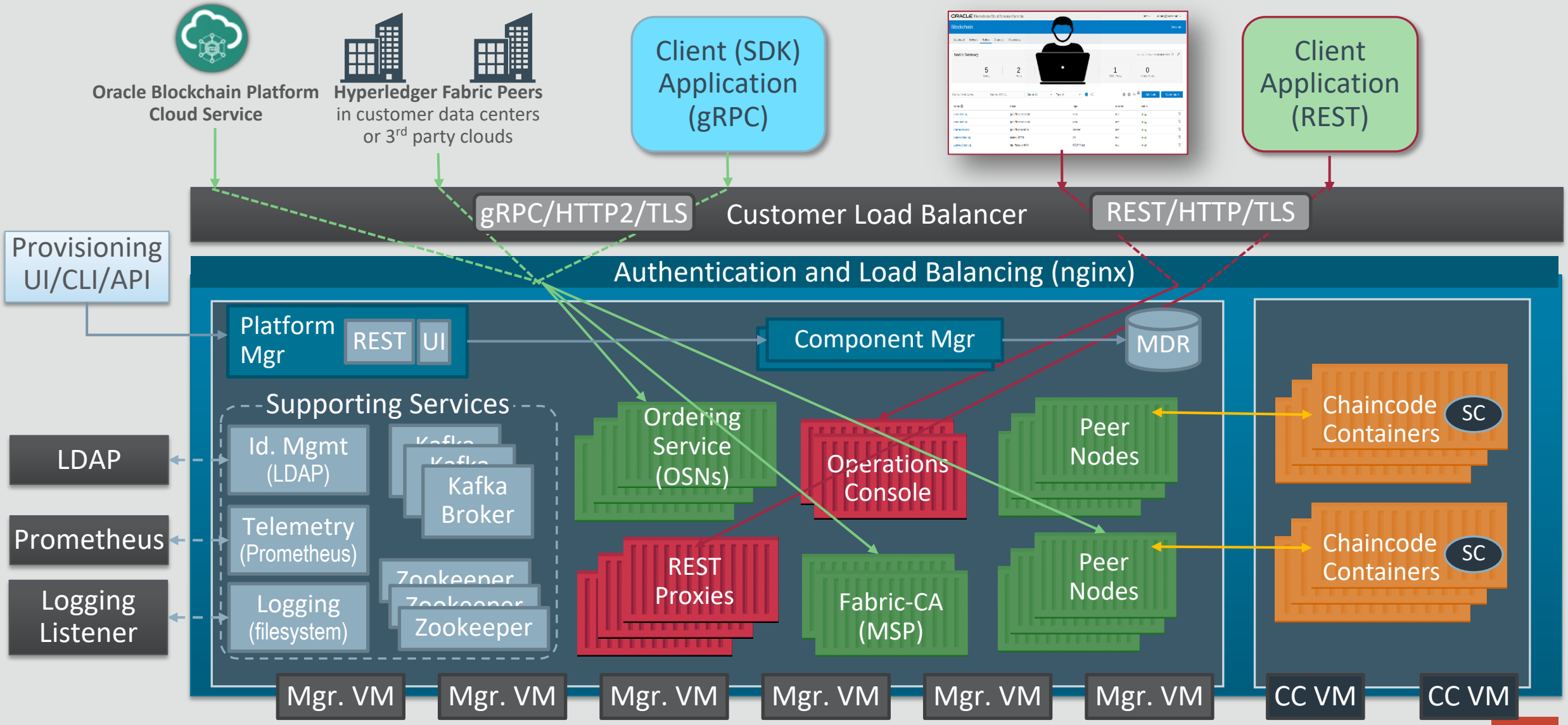
2. Oracle Blockchain Platform Console : managing, configuring, and monitoring each instance using Web UI or an extensive set of REST APIs

3. Oracle Blockchain REST Proxy : simple REST APIs to invoke transactions, query the ledger, and subscribe to events

4. OpenLDAP : out-of-the-box identity management and support for using an external LDAP Directory Service

Supported Hypervisors:
Oracle VirtualBox, versions 5.x and 6.0 or later
VMware vSphere ESXi, version 6.7 or later
Oracle Linux Virtualization Manager v4.2.8.2-1.0.8.el7 or later

OBP Enterprise Edition Architecture



Oracle Blockchain Platform - Delivering Enterprise Grade Capabilities

Development & Integration
- SQL-based rich queries
- Synchronous & async REST APIs
- Java, GO, and Node.js SDKs
- Transaction history and ledger replication in relational database
- Enhanced REST Proxy Clustering

Performance at Scale
- Parallel transaction execution
- Dynamic scale-out
- Much faster world-state DB

Supportability & Operations
- Dynamic configuration
- Monitoring dashboard
- Zero-downtime managed patching
- Autonomous recovery agents
- REST APIs for maint. and admin
- Full DR enablement

Security & Confidentiality
- Integrated identity management
- OOTB Integration with Directory Services
- Data encryption at-rest
- Certificate revocation management
- REST proxy SSO and multi-identity support
- Support for 3rd party intermediate CA

Deployment
- Choice of Virtualization Platform: VirtualBox or VMWare
- Preassembled, a few click deployment: Ready to deploy apps in mins
- High availability VMs

Oracle Cloud Feature

Enterprise Edition Feature

Thank you

Wonjo Yoo

Principal Solution Engineer
Cloud Excellence Team