쉽게 따라하는 블록체인과 DAPP개발 이론과 실천(Ethereum과 Hyperledger를 중심으로)

기술 -> 활용
특징 security / 투명성 / decentralize

4월 4일 목요일
1.블록체인 핵심기술
블록체인 소개 및 특징
블록체인 기술(퍼블릭 VS 프라이빗)
합의알고리즘(PoW, PoS, DPoS, PBFT)
Casper, Plasma, Sharding

2.Ethereum Smart Contract 소개 및 Dapp 개발
Smart Contract
Solidity 프로그래밍
DApp 개발기초
Ethereum 실습
Federated Learning

4월 5일 금요일
3.Hyperledger Fabric 핵심기술
Hyperledger Fabric 소개 및 구조
Hyperledger Fabric Functionalities
Hyperledger Fabric Model
Blockchain Network Identity, Membership, Peers, Private data Ledger

4.Smart Contracts and Chaincode
Use Cases

5.Hyperledger DApp 개발

1.블록체인 핵심기술

1) 블록체인 소개 및 특징

1세대 BTC (가상통화, 자산거래) -> 2세대 Ethereum (스마트계약-비즈니스자동화, 분산앱Decentralize) -> 3세대 다양한 플랫폼(scalability//interoperability,IoT support)

BlockChain
데이터 분산 저장 기술의 일종, DLT(분산장부기술, Distributed Ledger Technology)
저장된 데이터를 모든 사용자에게 분산하여 저장
block 단위의 데이터를 chain처럼 연결하여 저장
공개키암호화(RSA, 비대칭키암호화)기반 / 해시암호화기반(머클트리구조, 루트해시) / 디지털 서명
보안, 투명성, 분산화가 중요 특징

*중앙 서버가 없이 P2P로 투명하게 정보들을 다룰 수 있음 (중앙관리자를 얼마나 믿을 수 있는가, 신뢰할 수 있는가, client server VS P2P network)

**비트코인에서의 이중거래방지(double spending 방지)
금융거래에서 이중거래가 생기면 치명적 오류->이중거래 방지를 위한 다양한 합의알고리즘 개발
블록체인은 한줄이 되어야 함. 분기가 일어나면 이중거래 발생가능성이 생김.
경쟁에서 진 체인이 소멸되어야 이중거래 없어짐.
이때 합의 알고리즘을 통해서 경쟁에서 이기는 체인을 선택하게 됨
분기가 일어나지 않게 함으로 이중거래를 방지하는 것,
분기가 일어나도 분기를 잘 관리할 수 있는 합의 알고리즘을 적절히 사용하는 것이 Block chain에서의 중요한 문제
blockchain Trilemma : Scalability / Security / Decentralization

토큰 / 코인 : 토큰-Main-net X / 코인-Main-net O

2) 블록체인 기술(퍼블릭 VS 프라이빗)

퍼블릭 블록체인(Public Blockchain/Permissionless Ledger)
공개형 블록체인, 전 세계의 누구나 모두 읽고 거래 정보를 발송하고 거래가 유효한지 확인할 수 있으며, 누구나 합의 과정의 블록체인에 참여할 수 있음. 통상적으로 완전한 탈중앙화 시스템으로 여겨짐.
합의알고리즘 : PoW, PoS

프라이빗 블록체인(Private Blockchain/Permissioned Ledger)
폐쇄형 블록체인, 기관 또는 조직에서 권한을 통해 관리되는 블록체인을 말함. 이 해당 네트워크에 참여하기 위해서는 고유의 인증 방식을 통과해야 함.
합의알고리즘 : PAXOS, PBFT, Raft

3) 합의알고리즘 Consensus Algorithms

◇ PoW (Proof-of-Work, 작업증명) : **bitcoin**, ethereum, litecoin

+ : secure
- : slow throughput, expensive computations (BTC : 3-4tps, max 7tps)

조폐 과정에서 채굴자들에게 일을 했다는 것을 증명(작업증명)함으로 화폐의 가치와 보안을 보장하는 방식
풀기 어려운 문제를 빨리 해결한 사람에게 블록을 생성할 수 있는 권한을 주고 그 보상으로 코인을 제공
블록 거래 내용 변경을 위해 많은 자원이 필요해 위변조가 어려워 보안성이 좋음
이중지불 문제 해결(합의 알고리즘으로 PoW, Longest Chain 선택)
난이도 높아질수록 고사양 장비 필요, 전기 같은 리소스 낭비 심하고 속도가 느림

*비트코인 - 해시함수의 결과값이 특정값보다 작아지도록 하는 입력값(Nonce) 찾는 문제, Nonce 값 만드는데 SHA-256 알고리즘 사용 (256비트로 구성되어 64자리 문자열 반환하는 해시 알고리즘)
Nonce 값을 구해서 블록해시값을 구하고 이 블록 해시값을 식별자로 가지는 유효한 블록을 만들어내야함. (블록해시값을 0의 개수가 난이도를 의미함, 0이 많을수록 고난이도)
오픈소스 + **10년이 지났음에도 불구하고 해킹을 당하지 않음** / 블록체인 1세대

◇ PoS (Proof-of-Stake, 지분증명) : Dash

+ : Attacks more expensive, energy efficient
- : Nothing at stake, Prone to centralisation

PoW의 단점을 극복하기 위한 알고리즘 중 하나, 해시 파워가 많이 필요하지 않아 경제적.
*Nothing at Stake
블록체인이 포크되어 노드가 투표를 할 때 분기된 블록체인 두개에 모두 투표해도 전혀 손해보는 것이 없는 것, 따라서 포크문제를 해결하기 어렵다는 것.

◇ DPoS (Delegated Proof-of-Stake, 위임지분증명) : EOS, Ethereum Casper, Tendermint

+ : Cheap transactions, scalable, energy efficient
- : Partially centralized

특정 인원에게만 PoS할 수 있도록 권한 위임, 일종의 대표자가 생기는 것 -> 탈중앙화가 맞는지 애매함. 공격에 취약해짐
올바른 대표자 선출을 위해 본인을 밝혀야한다는 딜레마가 있음.
PoS에 비해 높은 확장성을 가짐, 거래 속도가 빠름.
Tendermint는 DPoS + PBFT 방식 사용

◇ PoA (Proof-of-Authority) : Ethereum Kovan

+ : Simple, Cost efficient, High throughput, scalable
- : Centralized

신분에 기반한 합의 알고리즘, 신원이 보장되어 있으므로 **private 네트워크**에 적합

◇ Paxos

Fault Tolerant Distributed System에서 여러 프로세스 간에 하나의 값에 동의하기 위한 프로토콜
leader를 선정하고 과반수의 동의에 의해 합의를 이룸

◇ BFT (Byzantine Fault Tolerance) : **Hyperledger**, Stellar

+ : High throughput, Transaction finality, Cost efficient, scalable
- : Centralized, Semi-trusted

*PBFT (Practical Byzantine Fault Tolerance) : Hyperledger
비잔틴 장군 문제를 해결하기 위한 실질적인 프로토콜 (단순 고장난 노드 뿐 아니라 **악의적인 노드가 있음으로 발생할 수 있는 문제를 해결해** 안정적으로 시스템이 돌아가도록 하는 프로토콜)
permissioned blockchain system에 적용
*FBA (Federated Byzantine Agreement) : Stellar, Ripple
*dBFT (Delegated Byzantine Fault Tolerance)

◇ RAFT (Replicated and Fault Tolerant)

PAXOS를 보완한 형태, 투표와 랜덤타임아웃을 통한 리더선출로 절차를 **단순화** 함
악의적인 노드는 고려하지 않음

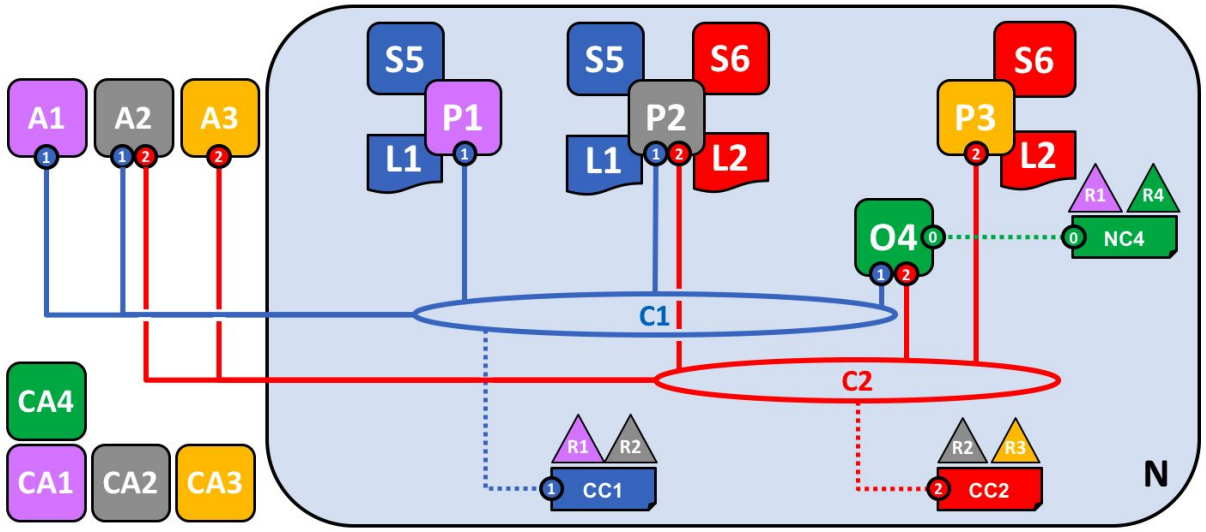
2. Blockchain Platform
1)Bitcoin
PoW방식
늦은 거래 처리속도, 큰 에너지 소모, 스크립트언어사용(튜링 불완전성, 반복문 제외 등 단점)
secure 뛰어남

2) Ethereum
GPU 기반 PoW -> PoS
스마트 컨트랙트 특화된 블록체인 플랫폼, 튜링완전언어(반복문 등 가능함)
분산 어플리케이션(DaPP : decentralized applications) 구현 가능
solidity를 사용한 smart contract로 구성

*확장성 문제 해결책 - Plasma, Sharding
Plasma : 블록체인 링커만 다른 곳에 두고 또 다른 체인(Child)을 만들자**(Off-chain Solution)**
Sharding : 한 줄이 아닌 여러 줄로 만들자, 전체 네트워크를 분할한 뒤 트랜잭션을 영역별로 저장하고 **병렬적으로 처리하는 방식 (On-chain Solution)**

3) Eos
DPoS 기반, DaPP 플랫폼 구축 가능, Ethereum보다 200배 빠른 처리속도를 가지고 있다고 함.
토큰으로 시작했다가 18년 6월부터 메인넷 런칭

4) Hyperledger Fabrics



- L : Ledger(블록체인[file system]과 state DB[=world state, levelDB]로 구성/privateDB를 가질수도 있음)
- S : Smart contract(s)(chaincode)
- P : Peer nodes
  L과 체인코드를 관리하는 노드, 다양한 피어 역할으로 수행
  endorser : 트랜잭션을 수행하고 승인
  committer : endorsements를 검증하고 트랜잭션 결과의 유효성 체크
- O : Ordering service(s)
  P와 A 사이에서 트랜잭션과 블록 전달, 트랜젝션 순서를 정하는 작업을 함
  *Ordering algorithms/Consensus Algorithms
  -SOLO (싱글노드방식)
  -Kafka (Crash fault tolerance / failed stop fault만 다룸, 악의적 fault는 못 다룸, 최소 3개 이상의 노드 필요)
- C : Channel(s)
  트랜젝션이 채널로 묶인 관련자들에게만 보이도록 하는 데이터 파티션 방법
  채널 안의 멤버들에 의해 합의과정이 이루어짐
- CA : Fabric Certificate Authorities

- A : Client Applications
- NC : Network Configuration
- CC : Channel Configuration

네개의 기관(R1, R2, R3, R4)이 협력하여 결정하고 동의에 서명함. Hyperledger Fabric network를 셋업하고 이용함. 각 기관은 CA를 소유함
R4가 네트워크 구성을 시작 - 네트워크 초기버전을 셋업할 권한을 부여
R4는 네트워크에서 비즈니스 트랜젝션을 수행할 의도는 없음
R1과 R2는 전체적인 네트워크 안에서 개인적인 통신을 필요함(R2와 R3도)
기관 R1는 채널 C1 안에서 비즈니스 트랜잭션을 수행할 클라이언트 애플리케이션을 가지고 있음.(A1, A2) / R2의 경우 C1과 C2 - A1, A2, A3 / R3의 경우 C2 - A2, A3
피어 노드 P1은 C1과 연관된 L1 관리 / P2-C1-L1 / P2-C2-L2 / P3-C2-L2)

네트워크는 NC4에 정의된 정책 규칙에 따라 관리, 네트워크는 R1, R4에 의해 관리
C1은 CC1에 명시된 규칙에 따라 관리, R1, R2가 채널 통제 (C2-CC2-R2,R3)
O4는 네트워크 N의 관리포인트이며 시스템 채널은 사용
O는 블록에서의 트랜잭션 순서를 배포할 목적으로 C1, C2를 지원

*Identity
PKI(Public key Infrastructure)는 전자인증을 발급하는 인증기관(CA)로 구성되어 신원리스트를 제공함
MSP(Membership Services Provider)는 이 중에서 누가 네트워크에 참여한 해당 기관의 멤버인지를 밝힘

**사용자들이 편하게 사용할 수 있게 UI/UX적인 부분이 개선되어야 함