암호화폐 서비스를 위한 pBFT 합의 알고리즘 기반

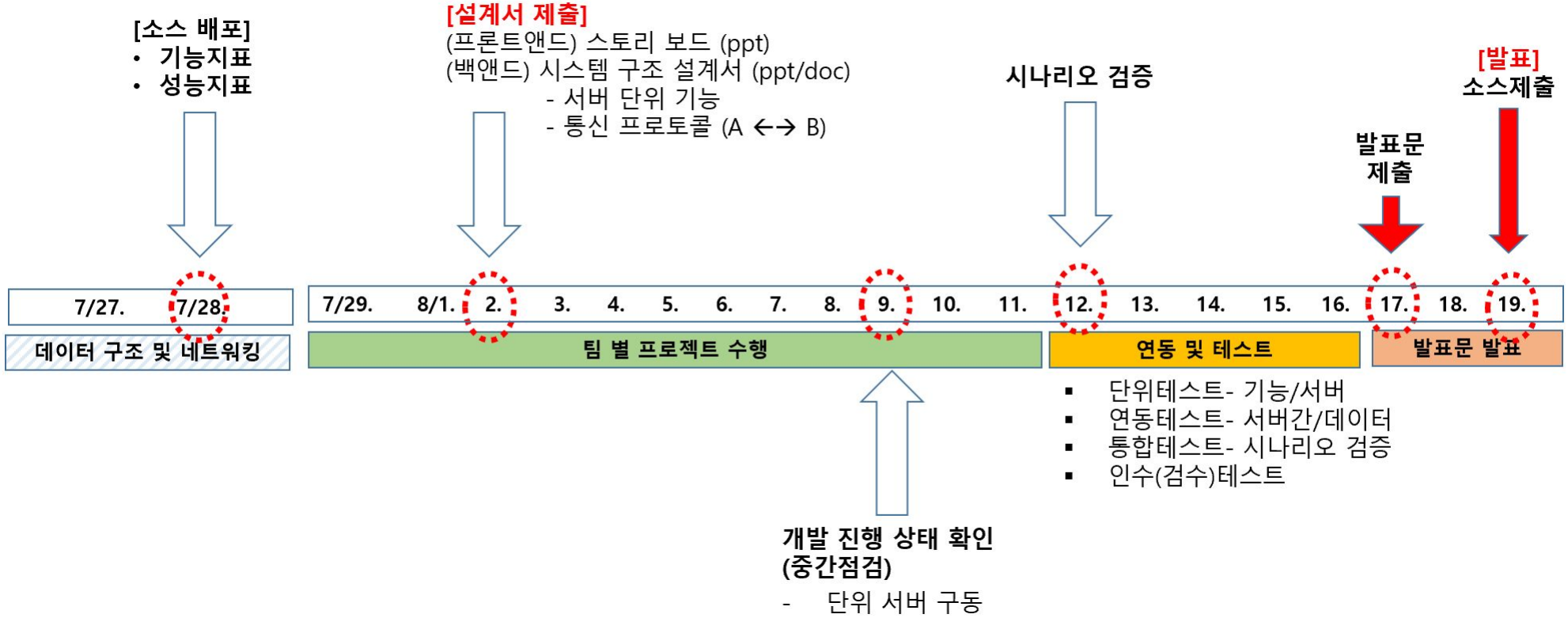
블록체인 코어 플랫폼 구현

블록체인 기반 핀테크 서비스 기술

A-Team

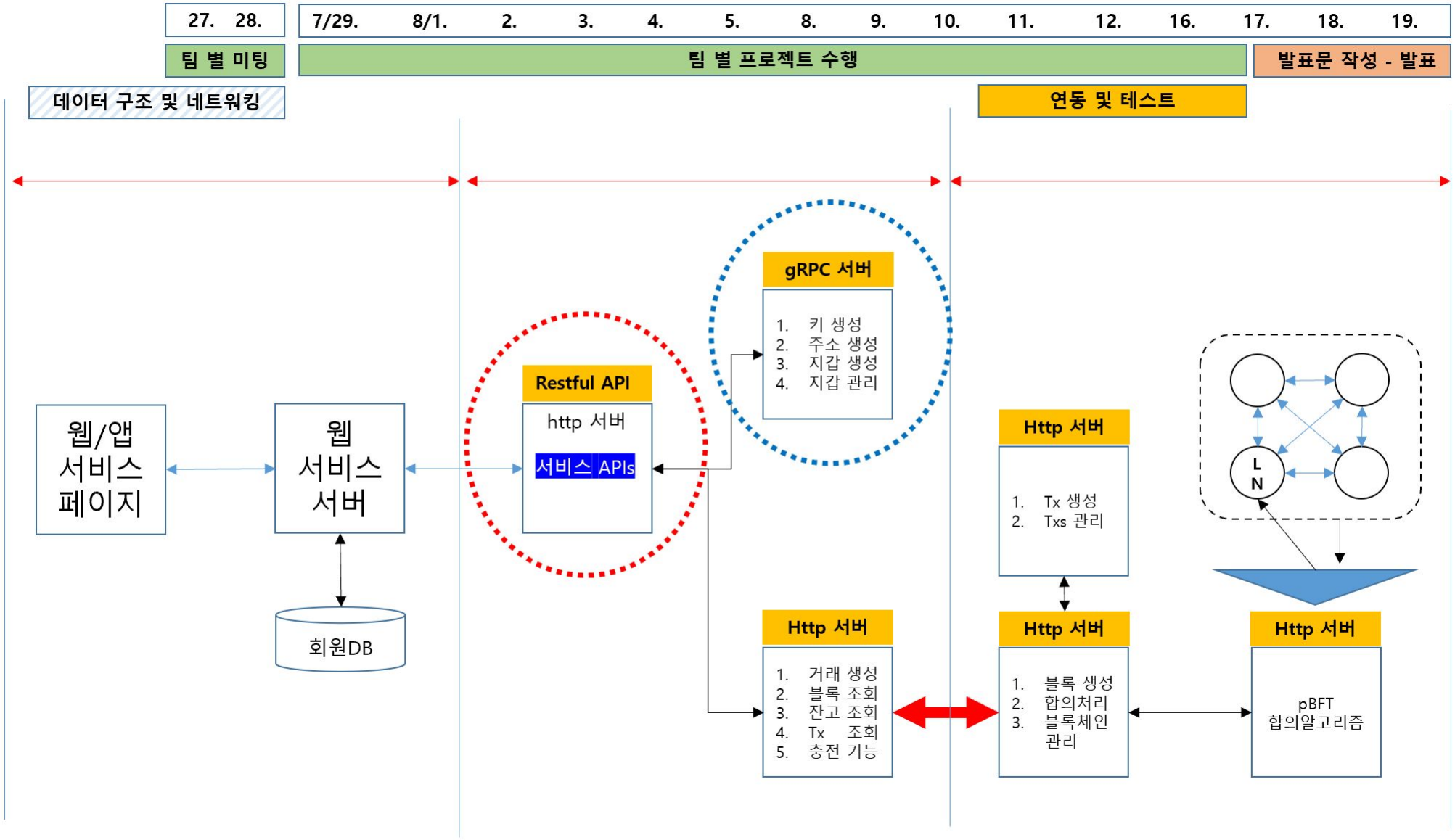
개발일정

-.진도대비 현황
-.진행 이슈공유



개발 목표

- 전체 시스템 구조
- Front-End vs Back-End 구분
- Team organization
- R&R in team



WBS

리더(수연) /스케줄러(수연) / 확인자(현수) / 취합자(수연, 현수) / 피드백(공통)

업무	담당자(정, 부)	기한	내용	진행상황	비고
분석					
설계서 제출	수연	8.2	FE-스토리보드(PPT) / BE-시스템구조설계서(PPT), 전체 아키텍쳐, 서버 단위 기능, 통신 프로토콜(A<->B)	완료	
구현					
API명세	수연	7.27		완료	
	중현, 현수, 수연, 지연, 동주	8.3	프로젝트 주제 선정 : POP Market		
	동주	8.3	엔티티-테이블 설계, 테스트	완료	
	중현		주문	완료	
	전체		DB	완료	
	수연		회원	완료	
웹-FE, BE	지연		상품	완료	
미들-RESTful server	전체	7.29	by Spring boot & Go	완료	
미들-HTTP서버(core통신)	현수		BlockCoreServer+API	완료	
미들-gRPC	현수	7.29	wallet생성, 관리, 지갑 주소 전달	완료	
코어-http 서버(tx)	수연		tx생성, 조회	완료	
코어-http서버(블록)	동주		블록 생성, 조회	완료	
코어--HTTP서버(합의)	현수		루트노드-전체 노드의 주소록 보유, 갱신, 공유	완료	
코어 - 합의노드(http통신)	전체		미들웨어 구축	완료	
테스트					
단위	각 구현자	8.9	중간점검 - 단위 서버 구동	완료	
	각 구현자		단위테스트-기능/서버	완료	
	수연, 현수		연동테스트-서버간/데이터	완료	
	수연, 현수		통합테스트-시나리오 점검	완료	
	전체		인수(검수)테스트	완료	
	동주, 현수		요청 10000개 기준 BPS측정 테스트 - 영상 기록(목표 : 합의 완료 노드 180~200BPS(/sec)	600/sec	
성능	동주, 현수	8.12-16	동시구동 노드 목표 50개(WINDOW기준)	완료(missed - 0.01%)	
발표					
발표문 제출	전체	8.17			
소스코드 제출	전체	8.19			

목차

1. Project Overview

POP Market - 가상화폐 기반 중고거래 플랫폼

: 중고시장의 사용자 = 잠재적 판매자

: 대부분의 거래 = 일회성이기 때문에 거래이력 = 거래 상대방의 신뢰성 가늠 척도

: if 거래이력의 무결성 보증 → 상품의 객관적 이력 확보

: if 가상화폐를 결제수단으로 입/출금행위의 중개자 역할 참여 → 개인간 거래 피해 방지

⇛블록체인을 활용, 사용자의 니즈가 사용자간 충족될 수 있는 플랫폼 실현

2. What is the problem

기존 중고거래 플랫폼의 문제점

- **객관적 신뢰 수단 부재** (∵ 거래 중재자 부재)

  → 중고 상품의 사용감, 시세 등 오직 판매자의 도덕성에 의존해 판단해야 하는 거래 시스템

2. What is the problem

기존 중고거래 플랫폼의 문제점

➡ 주요 3사 플랫폼 등 분쟁조정신청 현황 및 증가률

구 분	'20년	'21년	증감	분쟁 증가율
당근*켓	352	1,620	1,268	360%
중고*라	173	780	607	351%
번개*터	121	973	852	704%
기타	260	804	544	209%
합계	906	4,177	3,271	361%

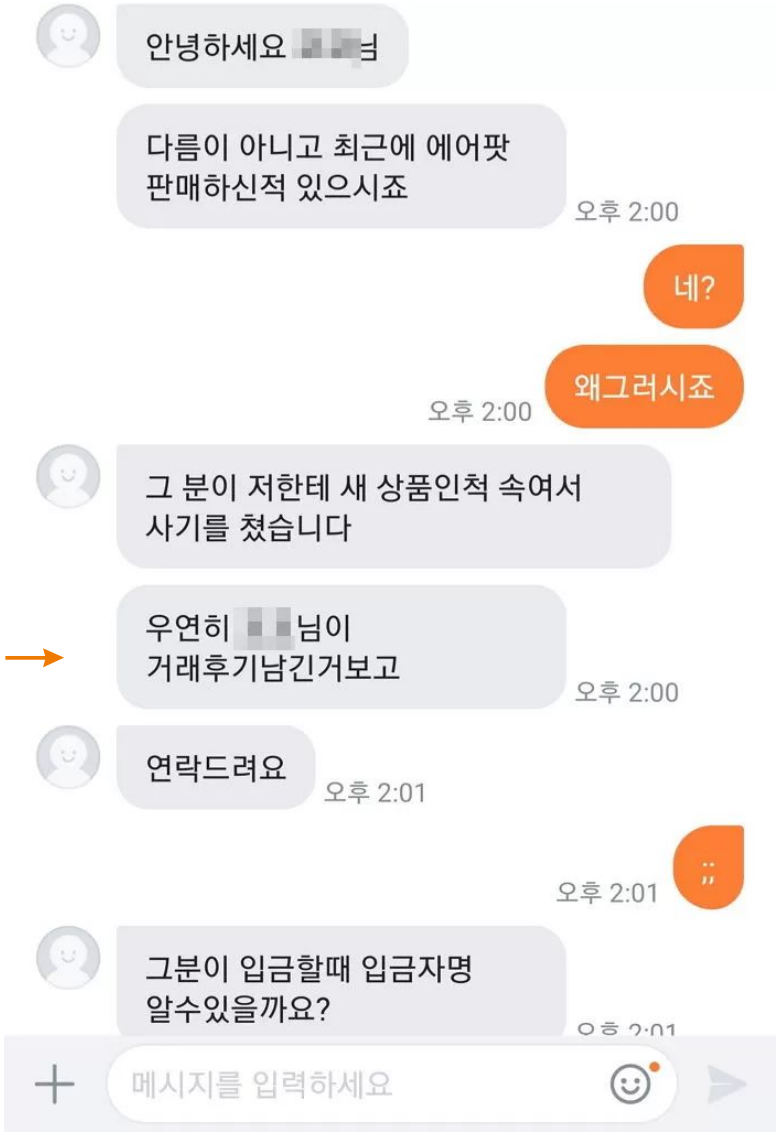
* 기타 : 카페, 밴드, 블로그, 카카오스토리, SNS(인스타, 페이스북, 트위터 등)

주요 3사 플랫폼 등 분쟁조정신청 현황 및 증가율 /KISA

2. What is the problem

기존 중고거래 플랫폼의 문제점

수정/삭제 가능한 정보에 의존할 수밖에 없는 구매자



3. What is needed

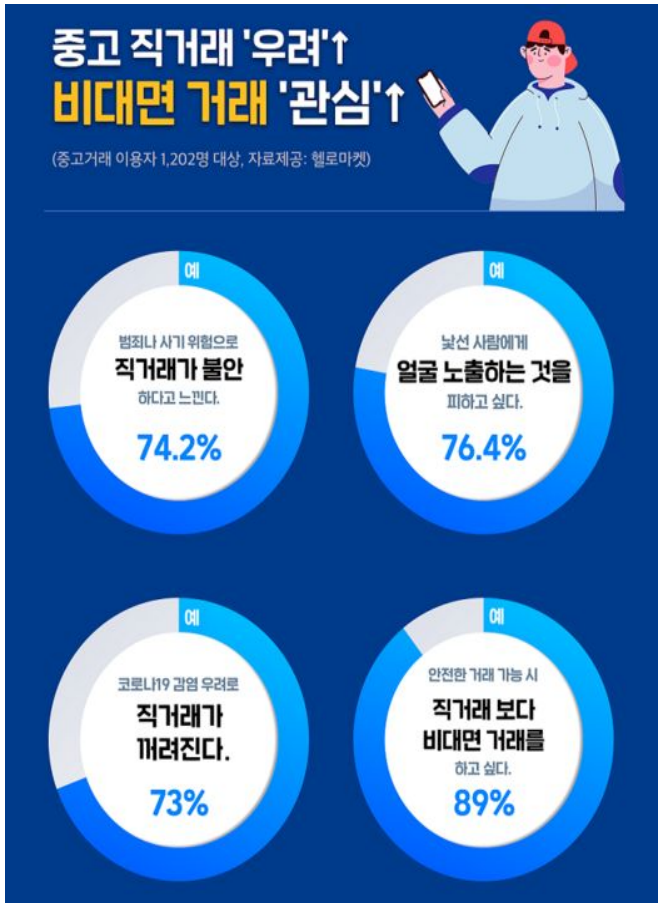
문제 해결 방안

1) 블록 체인 결제 시스템 도입(거래내역정보 공유)

- 구매자와 판매자 사이의 거래내역 정보를 저장 및 공개함
- 거래 계약서, 거래내역 정보를 공유함으로써 구매자가 판매자에 대한 신뢰성을 판단.

○ 블록체인 거래 구조

* 사용자의 모든 거래내역의 블록화
* tx에 sig값 삽입을 통한(혹은 tx hash값 생성시 walletid를 이용) 블록의 무결성 보증
* 합의 알고리즘에 따른 블록 생성과정의 안전성 보증



3. What is needed

문제 해결 방안

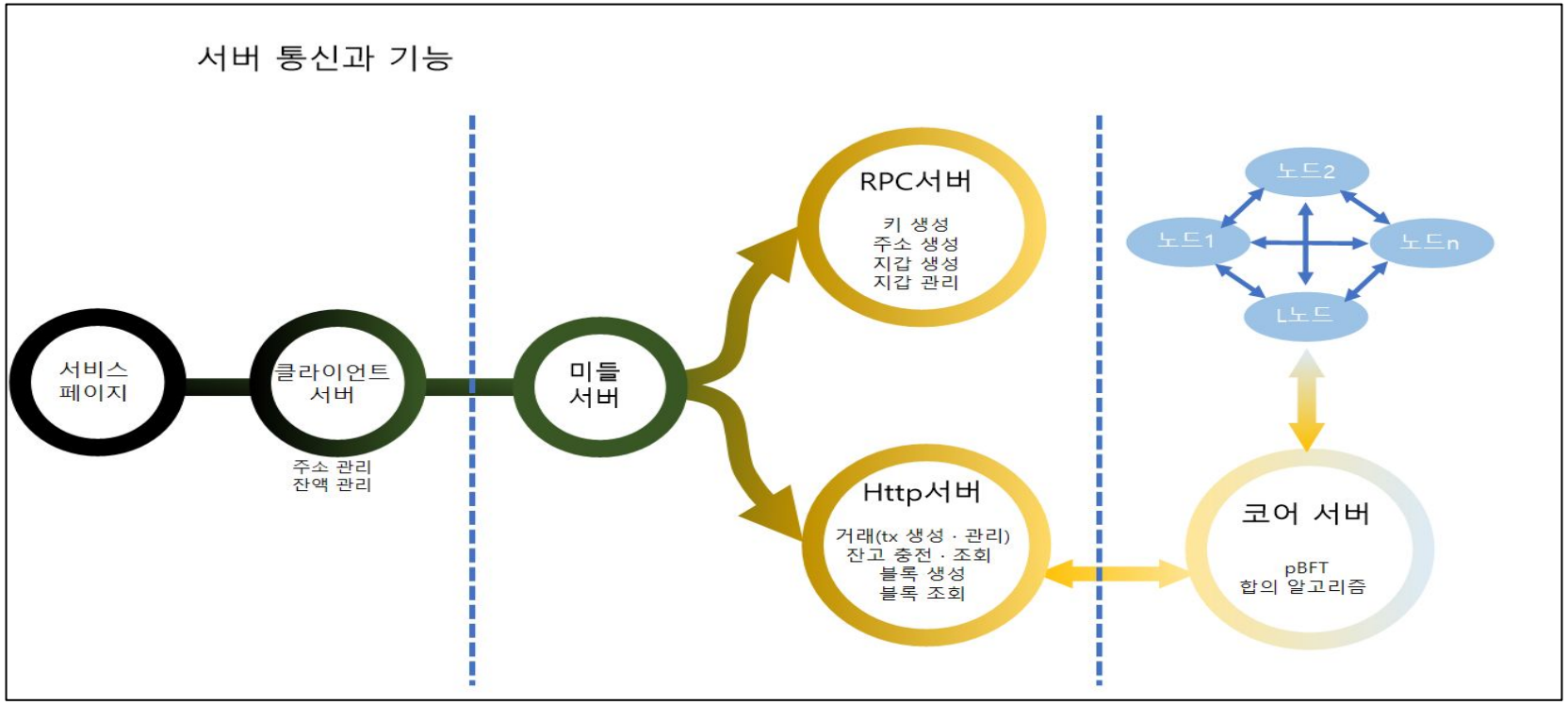
2) 입금 피해 사기 예방을 위한 후송금 - 구매확정 기능

- 구매 희망자가 입금 일시를 결정

→부도덕한 판매자가 돈을 편취하는 불상사 방지

4. What is our solution

◆ 블록체인 기반 중고거래 상점 서비스 네트워크 구조 및 설계



◆ 중고거래 특성상 지갑 호출 빈도 ↑ → 지갑관리를 RPC서버에서 하도록 설계

4. What is our solution

◆ 블록체인 기반 중고거래 플랫폼 설계서

==========

Layer1

==========

1. 회원
   - 1.1 회원 가입
   - 1.2 회원 암호 키 생성 (공개키, 개인키)
   - 1.3 회원 전자지갑 (wallet) ---------- A (create)
   - 지갑주소 생성
   - 잔고 확인 ---------- <---- DB

2. 거래
   - 2.1 상품 등록
   - 2.2 상품 조회 ---------- <----DB
   - 2.3 거래금 입금 ---------- <--- DB
   - 2.4 거래 확정 (tx 생성, blk 생성) ---------- B(create)
   - 2.5 거래 조회 (TX/BLOCK)
     - 2.5.1 서명 검증 후 Block 조회 ---------- C (ref)
       - 구매 이력
       - 판매 이력
       - 블록체인 메타 (header) 정보
       - tx 메타 정보 (Txid)
     - 2.5.2 단일 tx 조회
       - txid
       - 구매자 (from)
       - 판매자 (to)
       - 거래액 (amount)
     - 2.5.3 복수 tx 조회

3. 암호화폐
   - 3.1. 충전 ---------- <-- DB
     - 3.1.1 충전 거래 tx생성 ---------- D (create)
   - 3.2. 잔고 확인 <---------- <--- DB

Layer2

Layer2(FunctionNum)	routePath	functionName	Method
A Layer3(1, 2)	"/makeWallet"	makeWallet	POST
B Layer3(3, 5, 6)	"/newTx", "/newBlock"	newTx, newBlock	POST
C Layer3(4, 9)	"/getBlockOne"	getBlockOne	GET
D Layer3(4, 7)	"/getTx"	getTxOne	GET
E Layer3(4, 8)	"/getAllTx"	getAllTx	GET
F Layer3(3, 5, 6)	"charge"	Charge	POST

4. What is our solution

◆ 블록체인 기반 중고거래 플랫폼 설계서

==========

Layer3

==========

<RPC>
1. 키 생성(newKeyPair)

2. 지갑 생성
url : /makeWallet
parameter : alias, prvkey, pubkey
receive : alias
return : walletid
send : walletid

<HTTP>
3. 서명(타원곡선 전자서명 방식-ECDSA)
parameter : prvKey, 거래내용, (rand.Reader(=난수))
receive : prvKey, 거래내용
return : ecdsa.Sign결과값
send : ecdsa.Sign값

4. 서명 검증
parameter : pubKey, 거래정보 데이터, Sign결과값
receive : pubKey
return : ecdsa.Verify결과값(true/false)
send : true/false

5. 트랜잭션 생성
url : /newTx
parameter : from, to, amount
receive : 내용(사용자간 또는 충전거래 또는 탈퇴한 사용자의 addr)
return : txid
send : txid

==========

Layer3

==========

6. 블록 생성
url : /newBlk
parameter : prevHash, height, data, tx
receive : txid
return : block
send : ?

6. 트랜잭션 조회
url : /refTx
parameter : txid
receive : txid
return : tx
send : tx

7. 트랜잭션s 조회

8. 블록 조회
url : /refBlk
parameter : blockhash
receive : block
return : block, true or false
send : block, true or false

9. 충전 거래
url : /charge
parameter : { w http.ResponseWriter, re *http.Request}
receive(request) : amount, walletId
return : amount
send(response) : true/false

==========

Layer3

==========

```
src
    go.mod
    go.sum

    ├─block
        Block.go
        BlockChain.go
        ProofOfWork.go

    ├─httpServer
        blockCoreServer.go
        blockMiddleWare.go
        ConsensusServer.go
        HttpServerAPI.go
        TxCoreServergo
        WalletMiddleWare.go

    ├─main
        main.go

    ├─transaction
        Tx.go
        Txs.go

    ├─util
        utils.go

    └─wallet
        wallet.go
```

4. What is our solution

◆ **블록체인 기반 중고거래 상점 네트워크 구조**

1. 웹 서비스

- 웹서비스 단에선 사용자와의 사용자 인터페이스 콘솔로 일반 웹브라우저 사용 제공

- SpringBoot, JPA, MYSQL 등 스킬을 활용하여 사용자에게 웹 기반의 정보 제공

- 최종 코어와의 통신을 위해 Middle서버와 통신.

2. Middle 서버

- 웹서비스단과 코어서버의 인터페이스 역할 수행

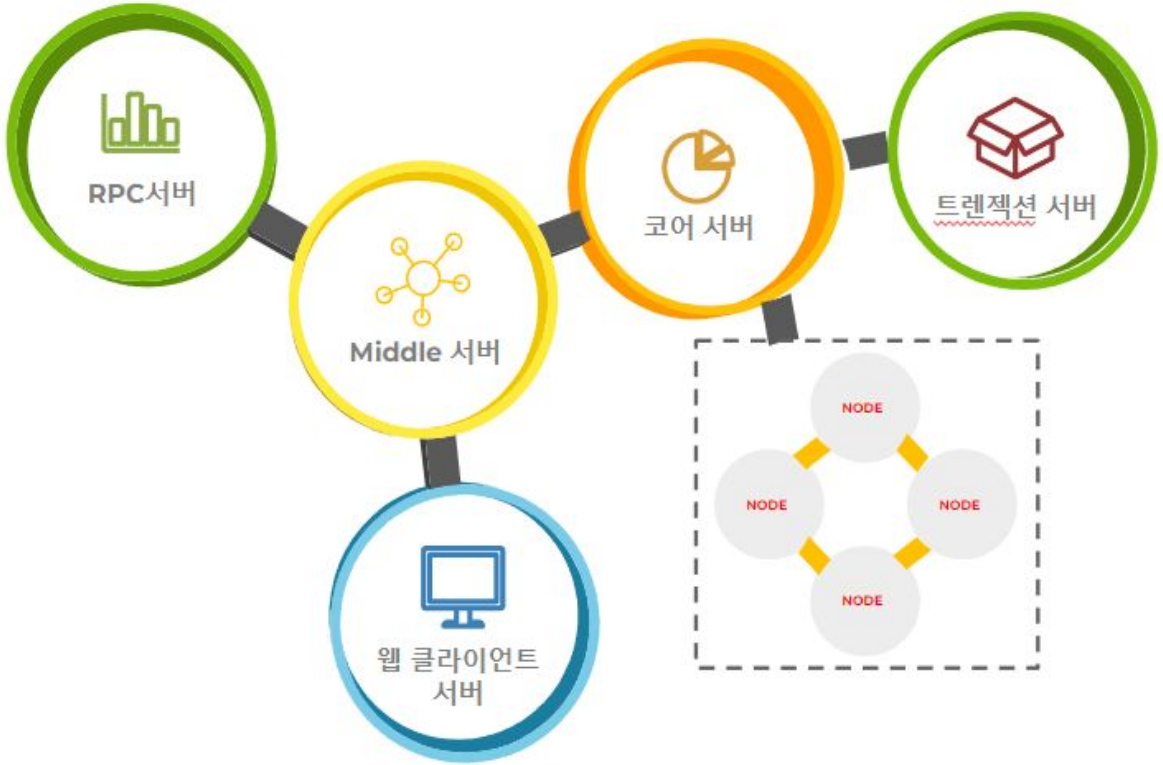
- 사용자가 요청한 서비스 정보를 RPC,HTTP서버로 요청

3. RPC서버

- 사용자들의 지갑관리 및 키 생성 및 정보 제공

4. 코어 서버

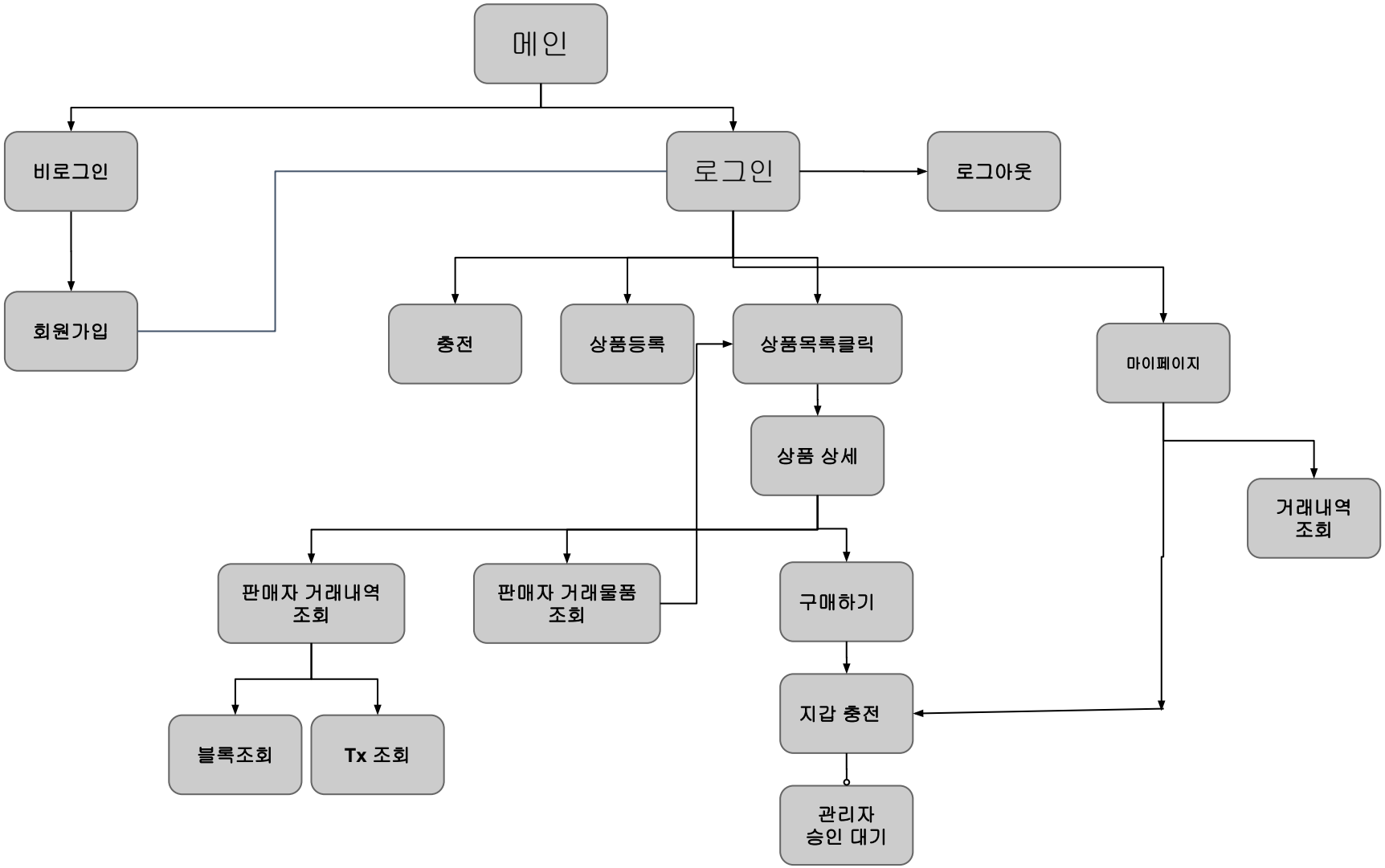
- 블록,TX 생성, 중고거래 플랫폼의 주요 거래내역 및 송금 및 결제 정보를 저장



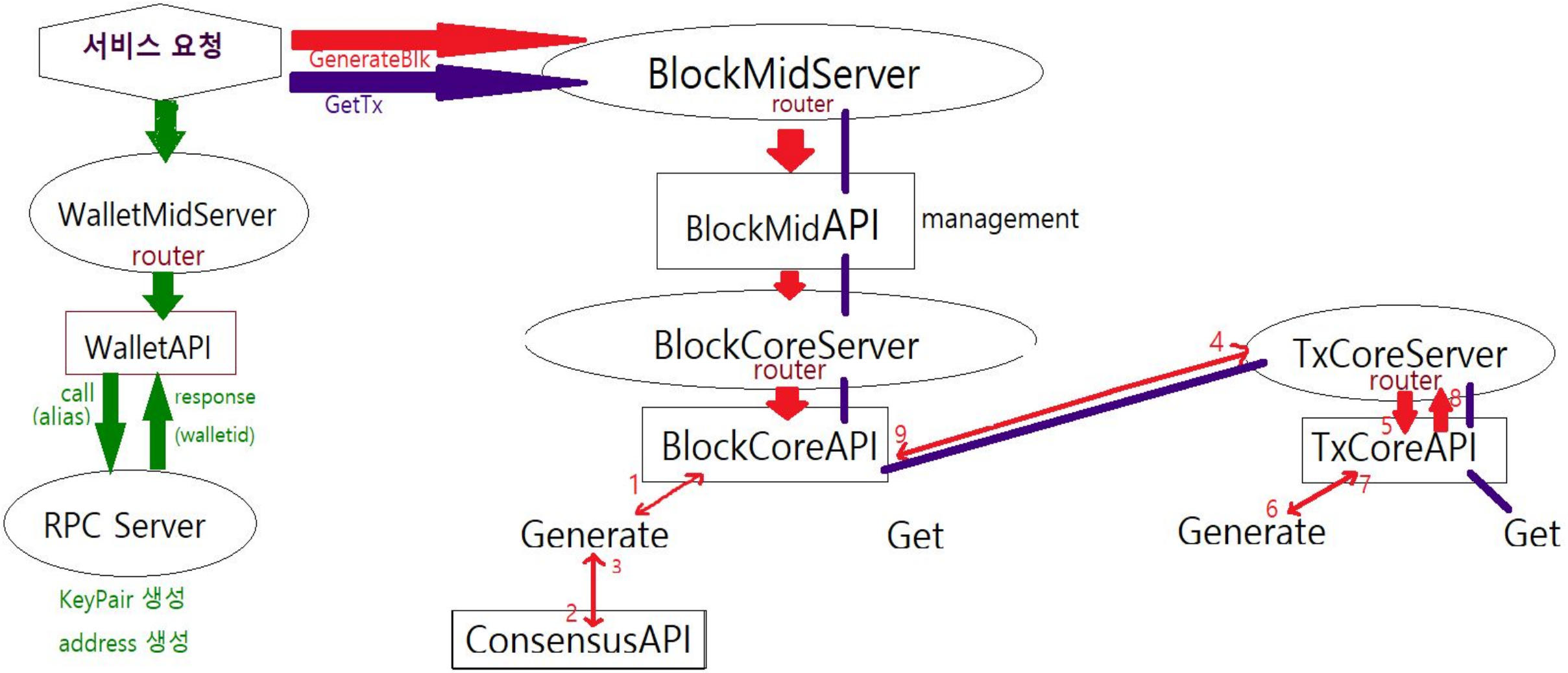
5. How it works : 개괄

클라이언트	컨트롤러	서비스	서비스 DB	RPC(지갑)서버	블록http서버 & 트랜잭션http서버 & 합의http서버
회원가입요청(GET, POST)	/member/register	registerMember()		:3030/MakeWallet요청후 walletid 받아와서 | walletid와 기본잔액 wallet에 저장 | 권한 정보 저장 | member에 저장	
회원로그인(GET)	/member / login				
메인 페이지 요청 (GET)	/, /product/list	getList()			
상품 등록 (GET, POST)	/product/register	registerProduct()	product에 저장		
상품 상세 페이지	/product/read	read() : 멤버와 상품서비스에서 정보 호출			
충전(GET, POST)	/wallet/chargeRequest		wallet에서 회원의 지갑정보를 가져와서 받은 금액과 함께 요청리스트로 등록		
충전 요청리스트 (GET)	/admin/chargeRequestList				
충전 승인 (POST)	/admin/chargeAccepted	chargeAccept()		GenerateBlock	NewTransaction | 노드간 합의 후 bl정보 반환 |받아온 정보에서 txid를 뽑아서 trade |받아온 블록DTO에 담아서 필요한 값만 chargeRequest리스트에 승인완료로 업데이트 |수신자 wallet 잔액과 송신자 잔액 업데이트
구매하기 버튼 클릭(GET)	/product/buy	read() 상품의 가격과 회원 지갑 잔액을 비교해서 충전 혹은 구매로 넘어가는 버튼보여줌			
구매하기 버튼 재클릭(GET)	/myPage/chargeConfirm	구매자와 판매자wallet 가져와서 trade에 저장 (confirm 0, txid 0)			
구매확정 버튼 클릭(GET,POST)	/myPage/tradeConfirm			GenerateBlock	NewTransaction | 노드간 합의 후 bl정보 반환
		| trade 업데이트 (confirm 1, txid 생성해서 담김)			

5. How it works : 서비스 도식화



5. How it works : 미들웨어-코어 도식화



5. How it works : Middleware 구현방식

1) HttpURLConnection(targetURL, RequestMap)

```
URL url = new URL(targetUrl);
HttpURLConnection conn = (HttpURLConnection) url.openConnection();
conn.setRequestMethod("POST"); // 전송 방식
conn.setRequestProperty("Content-Type", "application/json; charset=utf-8");
conn.setConnectTimeout(5000); // 연결 타임아웃 설정(5초)
conn.setReadTimeout(5000); // 읽기 타임아웃 설정(5초)
conn.setDoOutput(true); // URL 연결을 출력용으로 사용(true)

String requestBody = getJsonStringFromMap(requestMap);

@SuppressWarnings("unchecked")
public static String getJsonStringFromMap(Map<String, Object> map) {
    String jsonInString= "";
    JSONObject json = new JSONObject();

    for(Map.Entry<String, Object> entry : map.entrySet()) {

        String key = entry.getKey();
        Object value = entry.getValue();

        json.put(key, value);
    }

    return json.toJSONString();
```

6. Why it is efficient

★ pBFT 합의알고리즘 코어 네트워크에 대한 동작 성능 측정

이슈

- 요청 메세지가 한 번에 여러개 들어올 경우 합의가 되지 않는 불량노드가 많아짐
- 응답 보내는 도중 응답 완료가 와서 응답을 여러번 처리하는 문제 발생

해결

- 한 번에 여러 요청 메세지가 들어오는 경우 저장했다가 하나씩 처리하도록 버퍼 이용

성능

- 블록체인 처리 갯수 = 100개
- 소요시간 =3.07초
- 블록 개당 처리 소요시간 = 3.07초 / 100개 = 0.03초

6. Why it is efficient

★ 실시간 합의 노드 2개 다운 시 & 2개 추가 조인 시 합의 진행

이슈

- 합의 진행이 종료된 후에도 이전 합의 메세지를 처리하다가 노드가 다운 되도 합의 진행이 필요
- 노드가 다운됐다면 다시 추가하여 합의필요

해결

- 각 합의 진행 과정에서 해당 과정이 종료된 후에 이전 합의 메세지를 처리하지 않도록 조건 추가

성능

- 노드 두 개를 종료해도 합의가 진행되며, 두 개를 추가 조인해도 합의가 진행됨.

7. Who we are

Team Member

정현수

Middleware/Core Server

Java / Go

곽동주

Middleware/Back-end/Core

Java / Go

김중현

Service/Middleware

Java / Go

홍지연

Documentation/ Back-end

Java / Go

Team Leader

김수연

Service/Middleware/Core/Document

Java / Go

8. Future works

- 사용자 확보 : 결제수단(가상화폐)의 특성을 활용해 충전회차에 따른 적립금 제도 도입
- 상호작용 확보 : 사용자간 리뷰 기능 도입
- 능동성 확보 : 충전 승인 자동화