11주차 – 프로토콜 구현

계획 수정사항 - RSA인증,서명 관련부분 제외. 채굴을 통한 권한 부여로 대체

앞으로의 계획

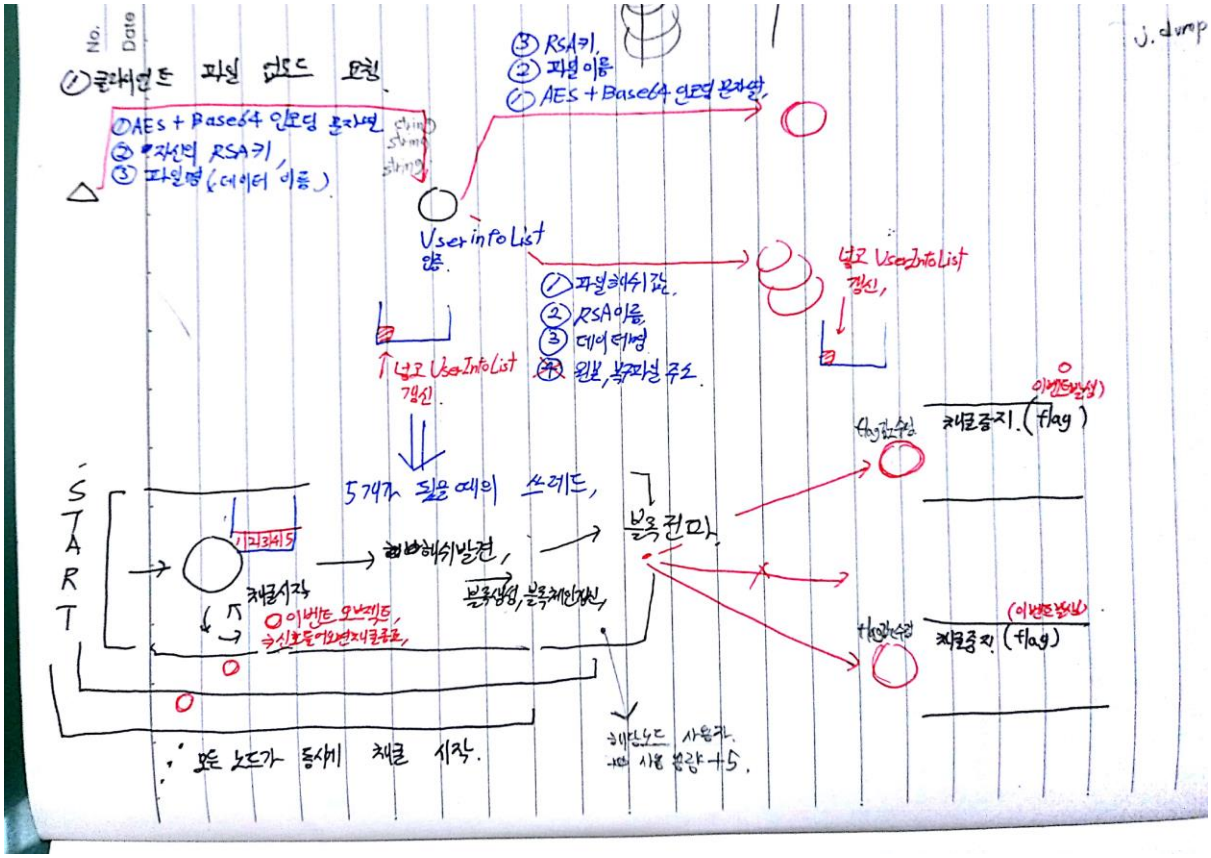
- 각자 맡은 프로토콜 JSON을 추가해서 프로토타입 완료하기
- 각자 맡은 프로토콜 세부 기능 추가하기.

3번 프로토콜 파일 업로드 과정(UploadFile) 구현 계획

-프로토콜 프로토타입 소스

https://github.com/bluepick3/BlockChain5/tree/master/ProtocolTest/UploadFile

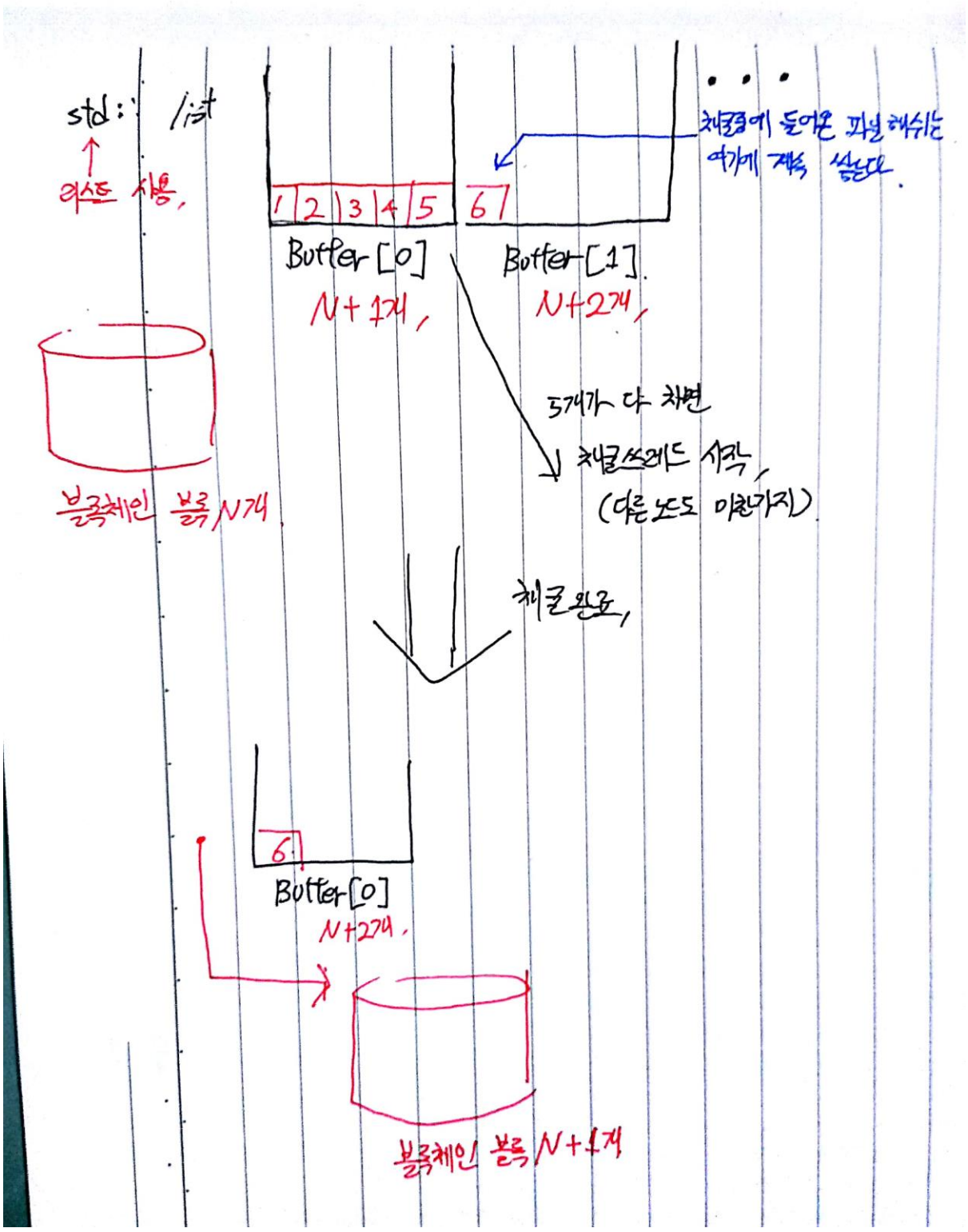
- 세부 기능 구현 계획



1. 파일 해쉬값들을 저장하는 HashBuffer 는 std::list로 구현된다. 여기서 Buffer[0]의 파일 해쉬가 5개가 되면 채굴 쓰레드를 생성하고 난스계산을 시작한다. 이 시점에서 다른 노드들도 모두 파일 해쉬가 버퍼에 5개가 찼을 것이므로 결과적으로 블록체인 망의 모든 노드가 동시점에 채굴 쓰레드를 생성하고 난스 계산을 시작하게 된다.

```
pair<string,string> findHash(int index, string prevHash, vector<string> &merkle) {
    string header = to_string(index) + prevHash + getMerkleRoot(merkle);
    unsigned int nonce;
    for (nonce = 0; nonce < 100000; nonce++ ) {
        string blockHash = sha256(header + to_string(nonce));
        if (blockHash.substr(0,2) == "00"){
            // cout << "nonce: " << nonce;
            // cout << "header: " << header;
            return make_pair(blockHash,to_string(nonce));
            break;
        }
    }
    return make_pair("fail","fail");
}
```

위 코드는 난스값을 찾아내는 코드이다. 위의 루프에서 "00"의 길이를 늘려 난이도를 높이고 nonce의 상한을 높인다. 또한 flag 값을 검사하는 코드를 넣어 만약 flag값이 변경된 것을 감지하면 난스 계산을 멈추고 채굴 쓰레드를 종료하도록 코드를 수정할 계획이다. 이 경우는 다른 노드에서 난스 값을 먼저 찾아내어 request를 보내왔을 때 벌어진다.



위 그림은 채굴 쓰레드 에서 난스 계산에 성공했을 경우의 처리 이다. (이때 HashBuffer 리스트는 블록체인의 블록의 연장선으로 취급되기 때문에 얼마든지 접근가능(사용가능) 하다)

Buffer[0]의 파일 해쉬 5개가 모두 채워졌기 때문에 채굴 쓰레드가 돌아가기 시작한다. 다른 네트워크에 위치한 다른 노드 들도 마찬가지 이다. 여기서 만약 난스 계산에 성공했을 경우 다음과

같은 절차를 수행한다.

1. 다른 노드에게 request를 보내 flag값을 변경시키고 채굴 쓰레드를 모두 종료시킨다.

2. 블록을 생성하고 나의 블록체인 갱신

3. HashBufferList 의 헤더, 즉 방금 채굴에 성공한 Buffer[0]를 제거한다.

4. 갱신된 블록체인을 다른 노드에게 전파한다. (어느 사용자가 채굴에 성공했는지 정보도 같이보낸다)

5.채굴에 성공한 사용자의 권한을 상승시킨다.

6.채굴 쓰레드를 종료한다.