**2021 NGP**

**Term Project 추진계획서**

제출일: 2021.11.07

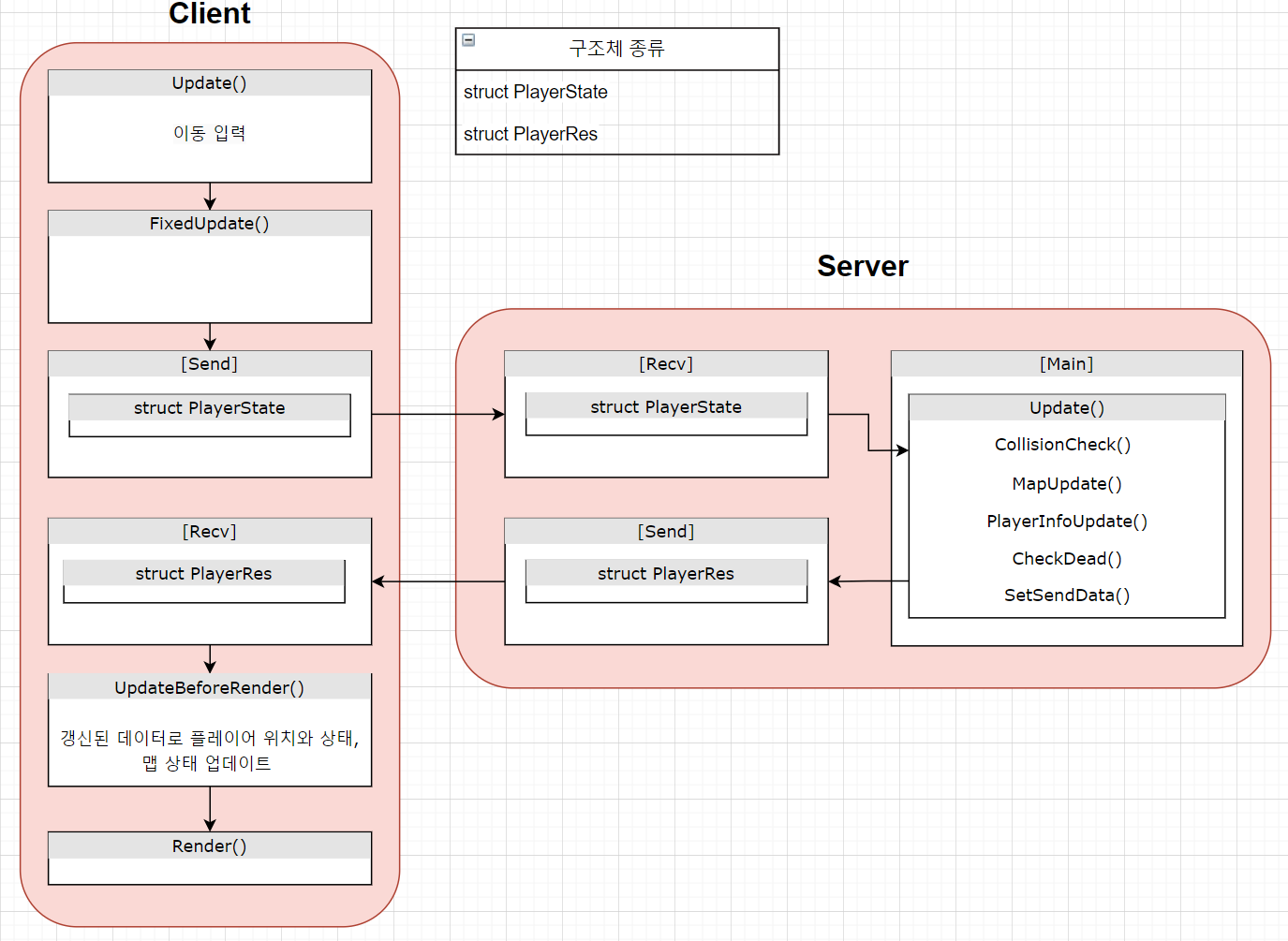
9팀

2018182021 윤성주

2018184033 최경훈

2015182020 안정인

1. **High-level design**
   1. 인게임 – 전투



1. 클라이언트는 Update()에서 받은 이동 입력 값을 저장해 놓는다.

->Update()에서 m\_tPlayerInfo구조체에 담긴 정보들(위치, 체력, 이미지 프레임)을 업데이트 한다.

1. 클라이언트는 구조체 PlayerState의 데이터를 서버에 Send 한다.

->Render()가 끝나고 SendRecvPlayerInfo()에서 자신의 플레이어 정보를 send하고, 서버로부터 모든 플레이어의 정보를 recv한다

1. 서버는 전달받을 데이터를 Recv 후, 각 플레이어들의 좌표와 상태를 이용해 맵의 아이템과 충돌 체크, 부활 여부, 맵 통과 여부를 서버의 메인 스레드에서 확인 후 처리한다.

->서버에서 CollisionCheck()에서 스킬들 좌표와 플레이어 좌표로 충돌체크 후 충돌을 했는지에 대한 여부를 판단한다.

1. 모든 처리가 완료된 후 PlayerRes 구조체에 서버에서 변화된 데이터를 담고, 이 구조체를 클라이언트로 Send 한다.

-> 서버에서 각 플레이어 정보를 tPlayersInfo[4]에 넣고, 클라이언트로 Send한다.

1. **low-level design**
   1. 인게임 – 전투
      1. Client Send

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 각 클라이언트는 SKILL\_TYPE, PLAYER\_STATE, TEAM\_TYPE 정보를 가진다.
2. PlayerUpdate()는 플레이어 좌표와, 플레이어 상태를 변경한다
3. PlayerInfo는 현재 좌표, 플레이어 상태, 팀 타입을 저장한다.

->싱글톤 클래스인 CDataMgr에 PLAYER\_INFO구조체 데이터를 저장한다

1. SendData()를 통해 서버에 구조체 전달한다.

-> SendRecvPlayerInfo()를 통해 서버에 구조체 정보를 넘기고, 모든 플레이어의 정보가 담긴 구조체인 STORE\_DATA 정보를 받는다

* + 1. Sever Recv & Send

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 서버는 각 클라이언트들에게 전달받은 데이터를 공유 데이터에 저장한다. 이는 크기가 4인 Vector로 저장되며, 각 클라이언트 인덱스로 벡터 인덱스에 접근한다.

-> g\_tStoreData.tPlayersInfo[4] 배열에 각 클라인트 정보를 담는다.

1. StoreData 구조체는 플레이어들의 좌표, 플레이어들의 스킬 좌표들, 플레이어들의 상태를 저장한다.
2. CheckCollision() 함수를 통해 저장된 공유 데이터로 충돌 연산을 진행하고, 충돌 여부를 저장한다.
3. SendData() 함수를 통해 충돌 여부와 공유 데이터를 담은 ResultInfo 구조체를 Send 한다.

-> ResultInfo구조체가 아님 STORE\_DATA 타입의 g\_tStoreData 구조체를 보낸다.

* + 1. Client Recv & Render

1. 서버로부터 전달받은 데이터를 ResultInfo에 저장한다.

-> CDataMgr의 m\_tStoreData에 저장한다

1. Render() 함수에서는 ResultInfo에 저장된 플레이어, 스킬들의 좌표를 이용해 자신과 다른 클라이언트들을 포함하여 모두 그린다.

->Recv -> Render 순서가 아닌, Render -> Recv로 변경했다

1. **팀원 별 역할분담** 
   1. 최경훈
      1. 클라이언트 초기
      2. 서버, 클라이언트 동기화 구현
      3. 플레이어 정보 Send & Recv
      4. 충돌 체크

