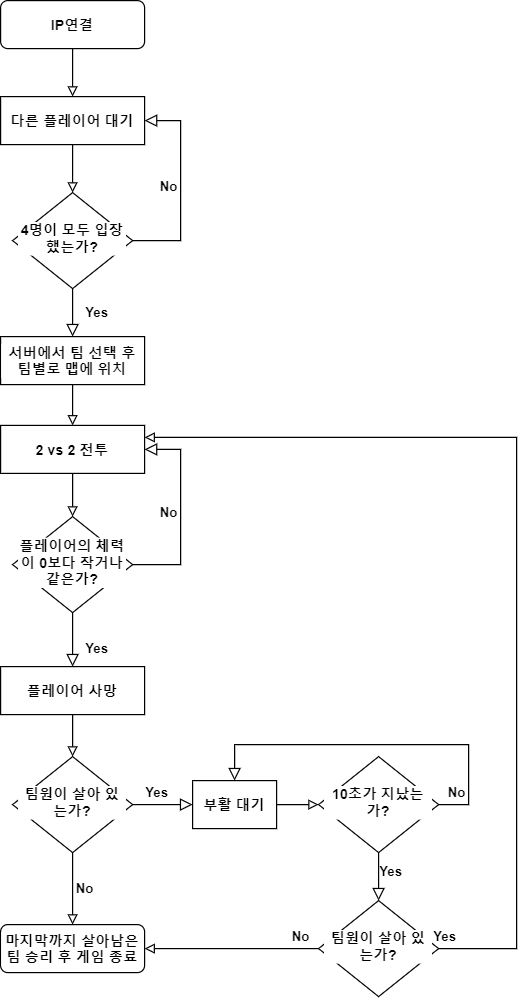
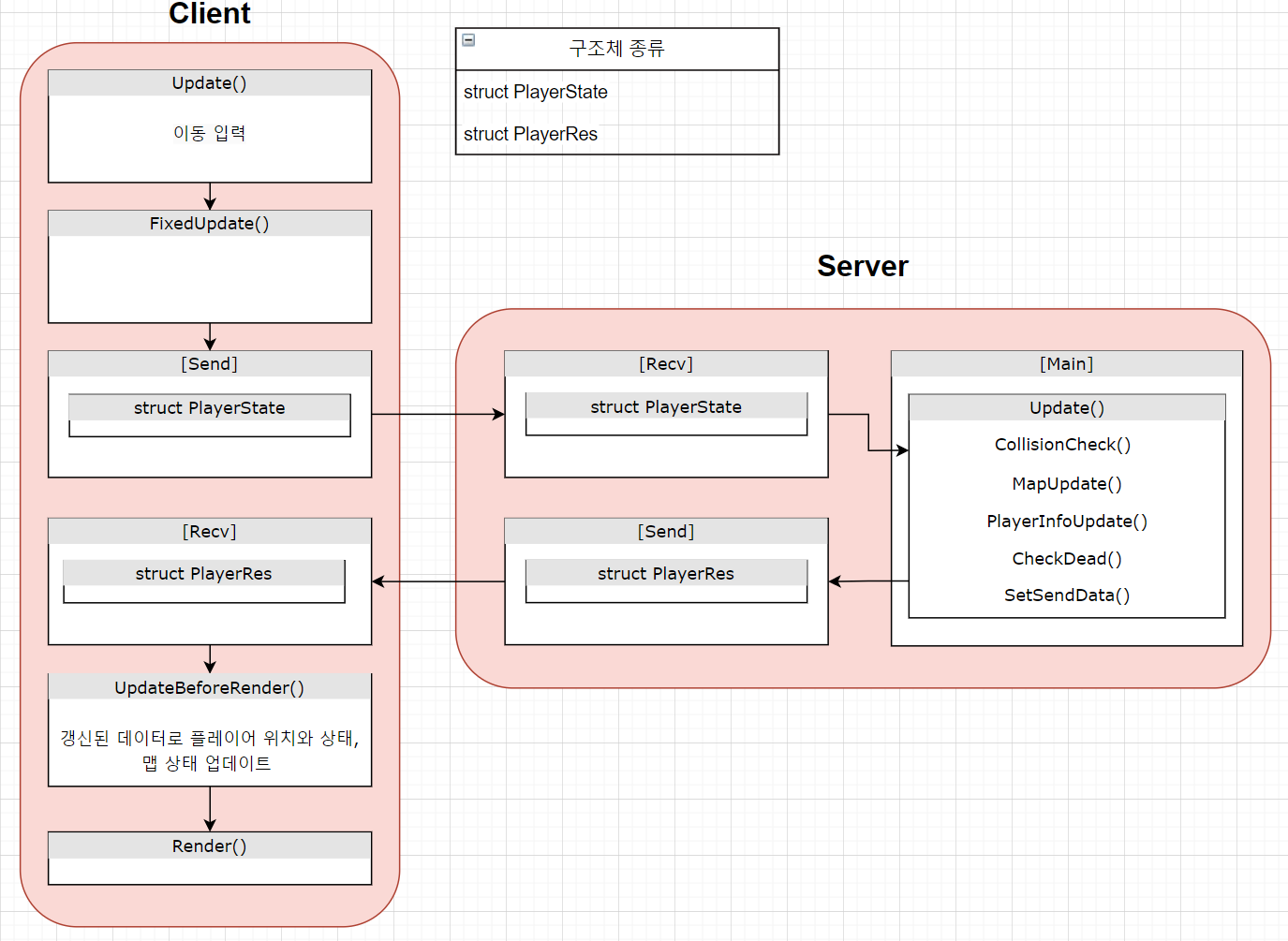
윤성주 기획서

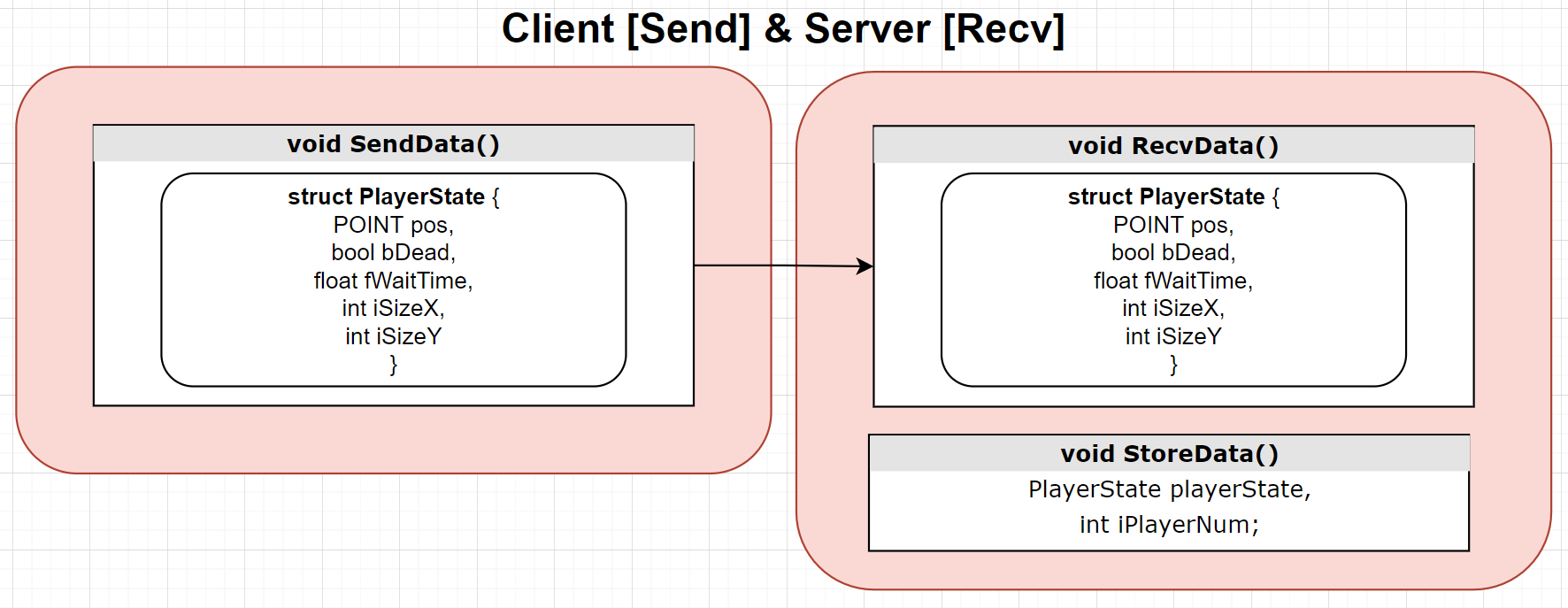
1. 하이레벨
   1. 전체 플로우차트



* 1. 함수 다이어그램



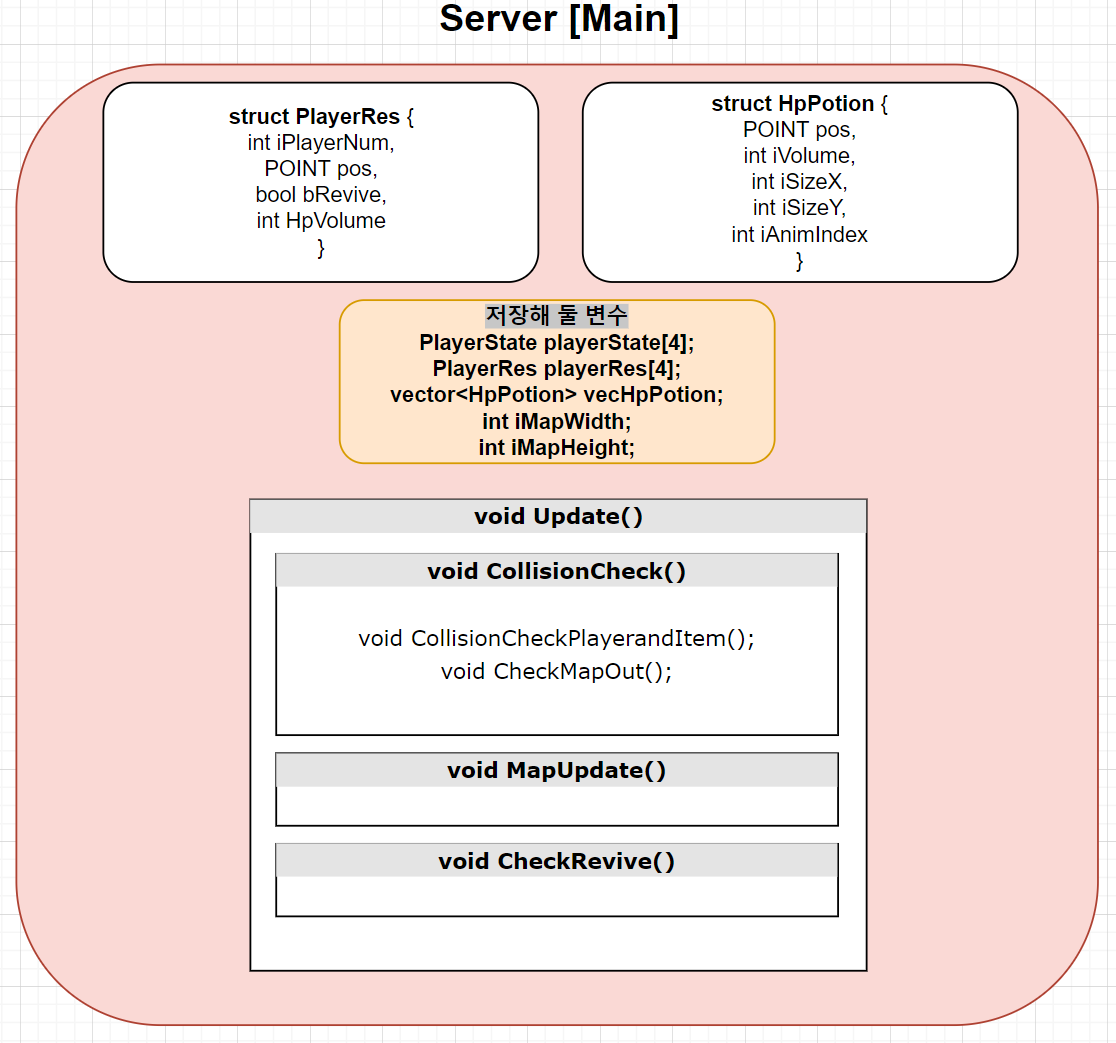
1. 클라이언트는 Update()에서 받은 이동 입력 값을 저장해 놓는다.
2. 클라이언트는 구조체 PlayerState의 데이터를 서버에 Send 한다.
3. 서버는 전달받을 데이터를 Recv 후, 각 플레이어들의 좌표와 상태를 이용해 맵의 아이템과 충돌 체크, 부활 여부, 맵 통과 여부를 서버의 메인 스레드에서 확인 후 처리한다.
4. 모든 처리가 완료된 후 PlayerRes 구조체에 서버에서 변화된 데이터를 담고, 이 구조체를 클라이언트로 Send 한다.
5. 클라이언트는 서버에서 보낸 PlayerRes 구조체의 데이터를 Recv 후, UpdateBeforeRender 함수에서 받은 데이터를 통해 플레이어, 맵의 정보를 업데이트 한다.
6. 모든 데이터가 갱신되면 Render한다.
7. 로우레벨
   1. Client – [Send] & Server – [Recv]



1. 클라이언트에서 Send할때는 구조체 PlayerState를 보낸다.

|  |  |  |
| --- | --- | --- |
| **PlayerState 구조체** | | |
| 인자 | 변수명 | 설명 |
| POINT | pos | 플레이어의 위치 |
| bool | bDead | 현재 죽어 있는지, 살아있는지 |
| float | fWaitTime | 부활을 위한 대기 타임 -> 10초가 넘고, 팀원이 살아있으면 부활. |
| int | iSizeX, iSizeY | 충돌체를 만들기위한 플레이어 가로, 세로 사이즈 |

1. 서버는 받은 구조체 PlayerState를 플레이어가 몇 번째 클라이언트인지 확인하여 StoreDate() 함수로 넘긴다.
2. StoreData() 함수에서는 서버의 전역 변수인 4명의 클라이언트의 PlayerState를 저장하는 배열에 iPlayerNum 인덱스에 받은 구조체 PlayerState 데이터를 넣는다.
   1. Server – [Main]



1. 서버는 구조체 PlayerRes 형태로 클라이언트에서 받은 데이터를 갱신한 정보를 저장해 두어 클라이언트에 Send한다.

|  |  |  |
| --- | --- | --- |
| **PlayerRes 구조체** | | |
| 인자 | 변수명 | 설명 |
| int | iPlayerNum | 4개의 클라이언트 중 몇 번째 클라이언트 인지 |
| POINT | pos | 갱신된 플레이어 위치 |
| bool | bRevive | 부활 여부 |
| int | iHpVolum | 체력약을 먹었다면 0보다 큰 값(체력 상승 값), 먹지 않았다면 0 |

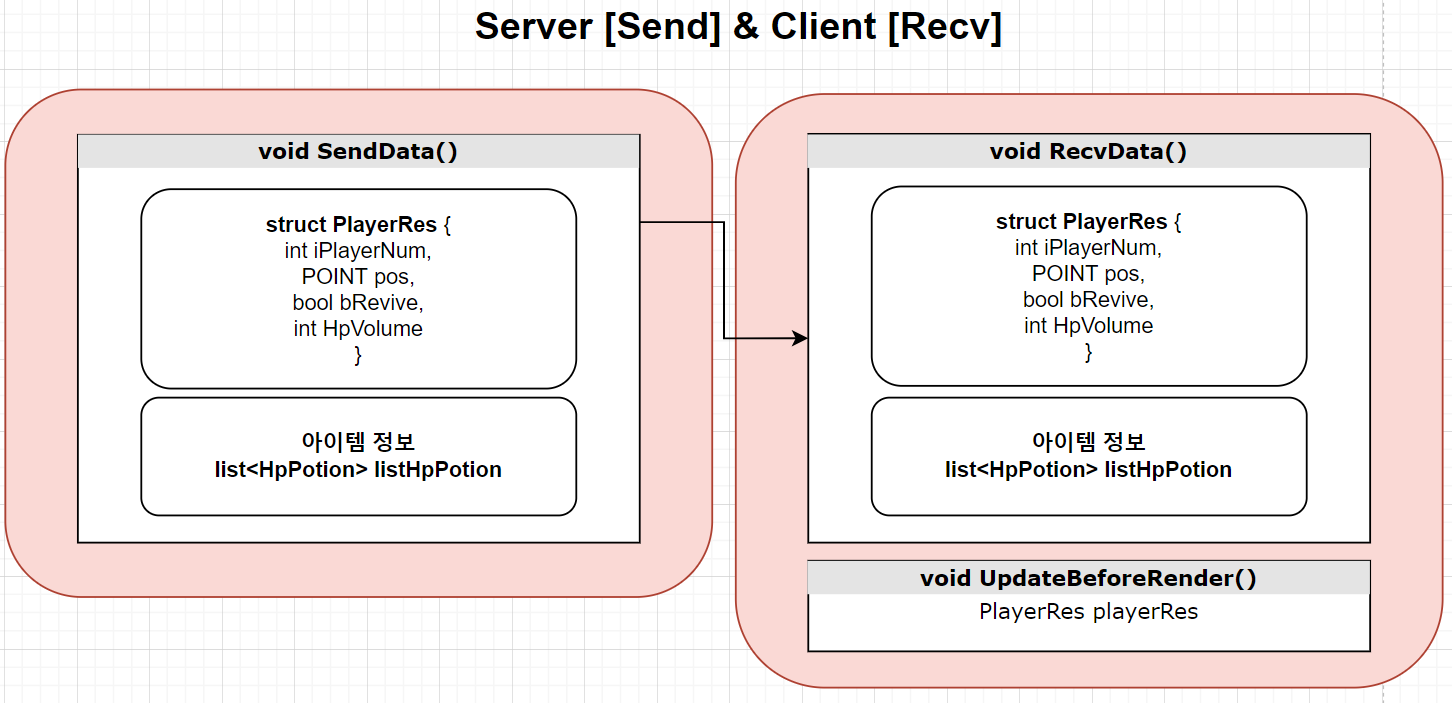
1. 서버에서 일정시간마다 HpPotion(체력약)을 맵에 생성한다.

|  |  |  |
| --- | --- | --- |
| **HpPotion 구조체** | | |
| 인자 | 변수명 | 설명 |
| POINT | pos | 위치 |
| int | iVolume | 용량 |
| int | iSizeX, iSizeY | 충돌체를 만들기위한 플레이어 가로, 세로 사이즈 |
| int | iAnimIndex | 애니메이션 순서 |

1. 서버에서 전역에 저장해 둘 변수 설명

|  |  |  |
| --- | --- | --- |
| 인자 | 변수명 | 설명 |
| PlayerState | playerState[4] | 클라이언트에서 받아온 데이터를 저장해논 배열 |
| PlayerRes | playerRes[4] | 클라이언트로 보낼 데이터를 저장해논 배열 |
| list<HpPotion> | listHpPotion | 맵에 존재하는 체력약의 리스트 |
| int | iMapWidth, iMapHeight | 맵의 가로, 세로 크기 |

1. CollsionCheck() 함수에서는 CollisionCheckPlayerandItem(), CheckMapOut()을 호출하여 각각 플레이어와 아이템 충돌 처리(충돌시 맵에서 삭제), 플레이어가 맵을 벗어날 경우 위치 재조정을 한다. 이때 전역변수를 사용하여 처리하므로 인자는 필요 없다.
2. MapUpdate() 함수에서는 체력약 생성등의 맵에 영향을 주는 일을 하여 맵의 상태를 갱신한다.
3. CheckRevive() 함수에서는 PlayerState 구조체 데이터를 통해 죽은 플레이어의 경우 대기시간이 10초 이상이면 부활하도록 한다. 부활 정보는 PlayerRes에 갱신된다.
   1. Server – [Send] & Client – [Recv]



1. 서버에서 갱신된 정보를 담은 PlayerRes 구조체를 연결된 클라이언트의 정보를 구조체의 첫번째 변수인 iPlayerNum으로 확인해서 그 클라이언트에 맞는 데이터를 Send 한다.
2. 아이템의 정보를 보내는데 이때 리스트를 보낼 수는 없으니, 리스트의 크기를 먼저 고정 길이로 보내고, 데이터를 가변 길이로 보낸다. 이때 HpPotion 구조체를 보낸다.
3. 클라이언트에서는 데이터를 받으면 UpdateBeforeRender()함수에 인자로 넘겨서 렌더링 전 갱신 작업을 수행한다. 갱신 작업에는 위치 재조정, 부활했을 경우 대기시간 초기화 및 부활, 체력약을 먹었을 경우 용량만큼 체력 상승 등이 있다.
4. 일정표

