네트워크 게임 프로그래밍

-Term 프로젝트 추진 계획서-

2018180044 최해성

2018182031 이지형

2019184036 홍명진

1. 애플리케이션 기획

1. 게임소개

게임: 스타크래프트 이미지를 사용한 갤러그

텍스트, 머리장식, 헬멧, 어두운이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명게임 컨셉: 2인의 플레이어가 비행기 모양의 플레이어 오브젝트를 조작하면서 아이템을 먹고 총알을 발사해 적을 쓰러뜨리면서 스테이지를 진행하고 보스몬스터를 잡으면 클리어한다.

타이틀

옅은, 장식이(가) 표시된 사진

자동 생성된 설명

스테이지

클리어

1. 조작방법

이동: W/A/S/D

공격: Enter

1. 몬스터 이동 패턴

* 교차형 
* 직선형 – 수직 빨간색이(가) 표시된 사진

  자동 생성된 설명
* 직선형 – 수평

텍스트, 사람들이(가) 표시된 사진

자동 생성된 설명

* 유턴형

실내, 꽃, 어두운, 밤이(가) 표시된 사진

자동 생성된 설명

* 계단형



* 고리형

텍스트, 가구이(가) 표시된 사진

자동 생성된 설명

* 특수형: 트랙터

여러개이(가) 표시된 사진

자동 생성된 설명

1. 보스패턴

* 분사형

옅은, 교통, 빨간색이(가) 표시된 사진

자동 생성된 설명

* 소용돌이형

옅은, 빨간색이(가) 표시된 사진

자동 생성된 설명

* 흩뿌리기

옅은, 교통, 빨간색이(가) 표시된 사진

자동 생성된 설명

* 흩뿌리기2

묶음이(가) 표시된 사진

자동 생성된 설명

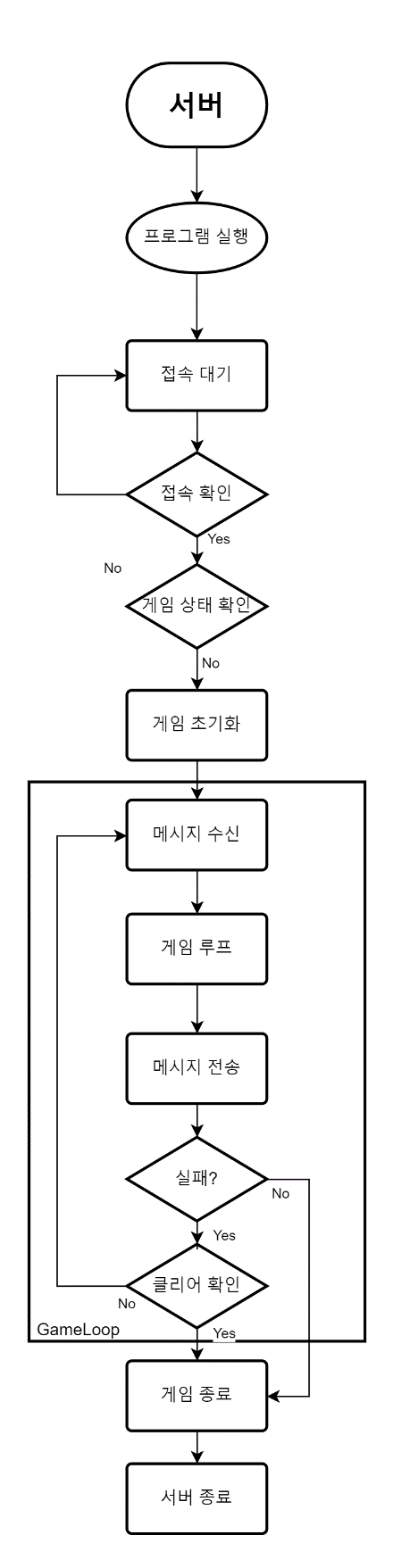
* 나선형

신호등, 군중이(가) 표시된 사진

자동 생성된 설명

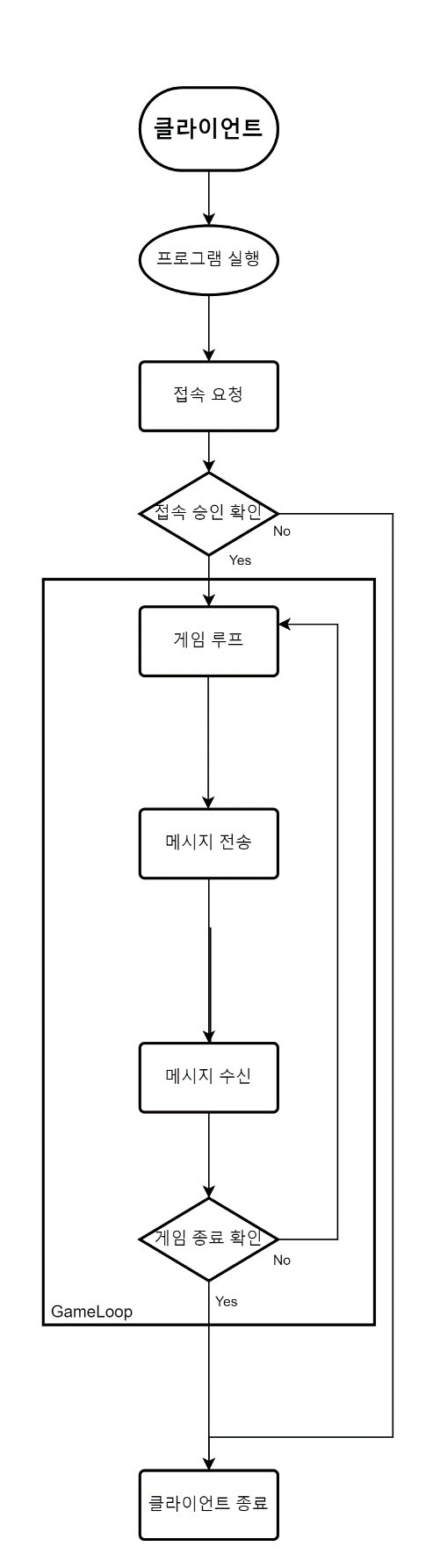
1. 서버
2. 클라이언트들이 서로의 플레이어 객체의 위치를 공유할 수 있게 한다.
3. 총알 & 몬스터의 충돌을 확인하고 몬스터의 상태를 업데이트 한다.
4. 몬스터들과 총알들의 위치를 클라이언트끼리 같게 한다.
5. 플레이어 & 몬스터의 충돌을 확인한다.
6. 플레이어 & 아이템의 충돌을 확인한다
7. 게임의 상태를 클라이언트끼리 같게 한다.
8. 아이템을 생성한다.
9. 클라이언트
10. 플레이어 상태를 업데이트 한다.
11. 클라이언트 사용자의 키 입력을 받아 처리한다.
12. 사운드를 재생한다.
13. 서버로부터 오는 메시지를 처리한다.
14. 서버로 플레이어 객체의 위치정보를 전달한다.
15. 화면을 렌더링한다.

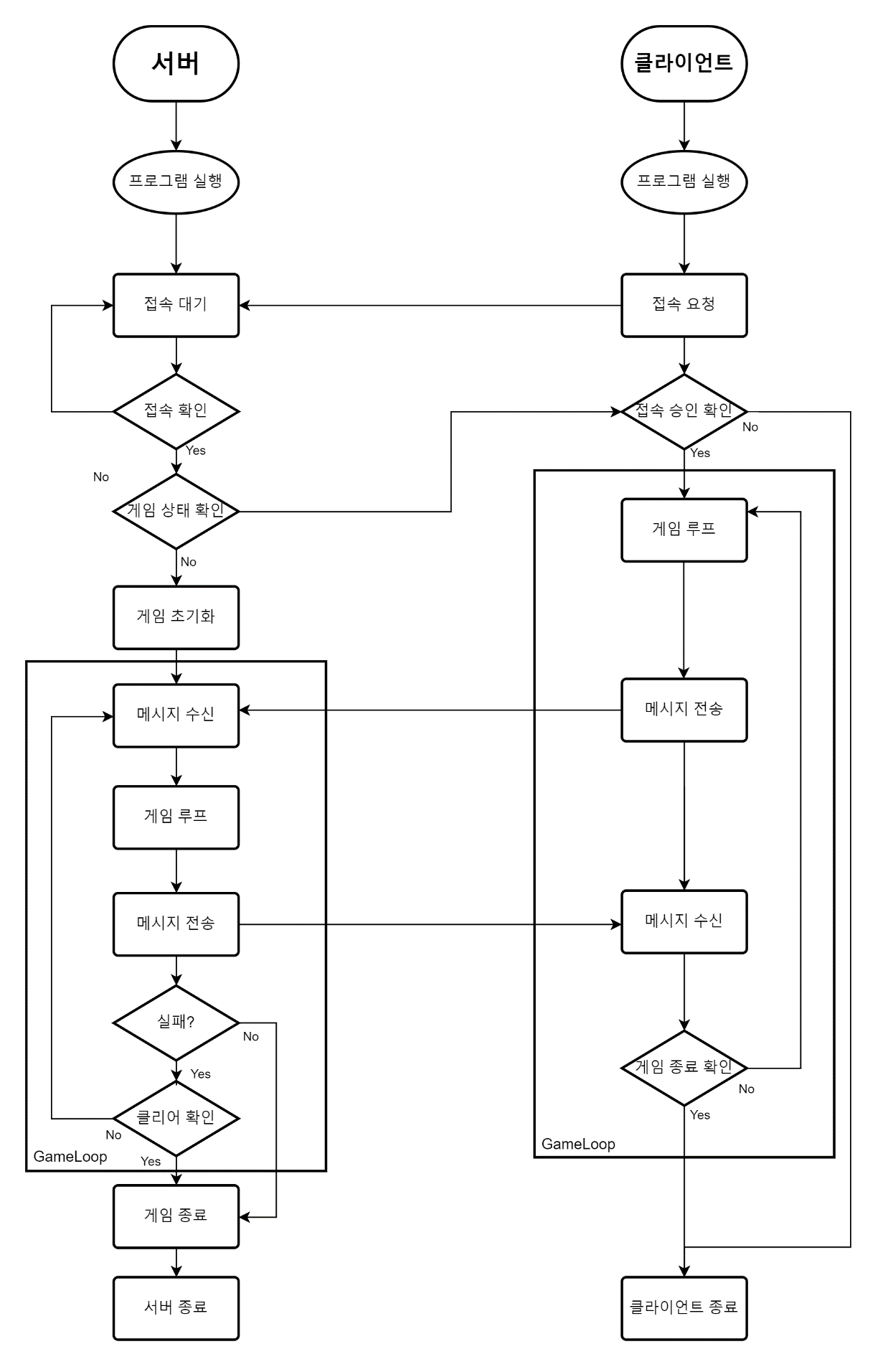
2. High Level 디자인

1. 서버에서의 Data Flow

서버는 프로그램이 실행되면 윈도우 소켓을 초기화 하고, 대기 소켓을 생성해 클라이언트의 접속을 대기하다가 클라이언트가 접속 요청을 하면 현재 게임 상태를 확인해서 게임 초기화를 합니다. 진행중인 게임이 없으면 새로운 게임을 시작하고 이미 진행중인 게임이 있다면 새로운 플레이어를 진행중인 게임에 첨가시킵니다. 서버는 클라이언트로부터 메시지를 받으면 받은 데이터를 이용해 게임을 업데이트하고 업데이트 된 데이터를 클라이언트로 전송합니다. 모든 플레이어가 사망해 게임이 종료되거나 보스 몬스터를 잡아 게임이 클리어 되면 서버를 종료합니다.

1. 클라이언트에서의 Data Flow

클라이언트는 프로그램이 실행되면 윈도우 소켓을 초기화하고 소켓을 생성해 서버에 접속을 요청합니다. 서버로부터 접속 승인 메시지를 받아 접속이 승인되었다면 게임루프를 실행하고 접속이 승인되지 않았다면 프로그램을 종료합니다. 접속을 승인 받아 게임루프에 진입한 클라이언트는 서버에 클라이언트의 정보를 담은 메시지를 보내고 서버로부터 결과 메시지를 받아 게임 종료를 확인해 게임이 종료되었을 경우 클라이언트를 종료하고 게임이 종료되지 않았을 경우 다시 게임루프를 돌게 됩니다.

1. 서버 & 클라이언트에서의 Data Flow
2. 서버 모듈

|  |  |
| --- | --- |
| 모듈 이름 | 기능 |
| CCore | 전반적인 프로그램의 흐름을 관리합니다. |
| CTimer | 프로그램의 시간을 동기화합니다. |
| CSceneManager | 씬을 관리합니다. |
| CScene | 씬의 세부정보를 관리합니다. |
| CObjectManger | 게임에 필요한 객체들을 관리합니다. |
| CObject | 게임에 들어가는 객체들의 세부정보를 관리합니다. |
| CItem | 게임 내 아이템들을 관리합니다. |
| CBullet | 게임 내 총알들을 관리합니다. |
| CNetworkDevice | 클라이언트와 메시지를 주고 받으며 메시지를 해석합니다. |
| CMessageDispatcher | 메시지에 따라 객체들에게 메시지를 전달합니다. |
| Telegram | 메시지의 세부정보를 관리합니다. |

1. 클라이언트 모듈

|  |  |
| --- | --- |
| 모듈 이름 | 기능 |
| CCore | 전반적인 프로그램의 흐름을 관리합니다. |
| CTimer | 프로그램의 시간을 동기화합니다. |
| CSceneManager | 프로그램의 시간을 동기화합니다. |
| CScene | 씬을 관리합니다. |
| CObjectManger | 씬의 세부정보를 관리합니다. |
| CObject | 게임에 필요한 객체들을 관리합니다. |
| CPlayer | 플레이어의 세부정보를 관리합니다. |
| CMonster | 몬스터의 세부정보를 관리합니다. |
| CBoss | 보스몬스터의 세부정보를 관리합니다. |
| CTractor | 트랙터의 세부정보를 관리합니다. |
| CItem | 아이템의 세부정보를 관리합니다. |
| CBullet | 총알의 세부정보를 관리합니다. |
| CSoundManager | 게임 내 사운드를 관리합니다. |
| CNetworkDevice | 서버와 메시지를 주고 받으며 메시지를 해석합니다. |
| CMessageDispatcher | 메시지에 따라 객체들에게 메시지를 전달합니다. |
| Telegram | 메시지의 세부정보를 관리합니다. |

3. Low Level 디자인

1. 서버 모듈

|  |  |  |
| --- | --- | --- |
| CCore | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | CTimer, CSceneManager, CObjectManage, CNetworkDevice를 초기화합니다. |
| Collision | void | CSceneManger의 Collision함수를 호출한다. |
| Update | int | CSceneManger의 Update함수를 호출한다. |
| LateUpdate | int | CSceneManger의 LateUpdate함수를 호출한다. |
| Logic | void | 클라이이언트로부터 메시지를 받고 처리한 후 타이머를 업데이트하고 게임 루프를 실행한 뒤 클라이언트로 메시지를 보낸다. |

|  |  |  |
| --- | --- | --- |
| CTimer | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 타이머를 초기화한다. |
| Update | void | 프레임 타임을 계산한다 |
| GetDeltaTime | float | 프레임 타임을 반환한다. |

|  |  |  |
| --- | --- | --- |
| CSceneManager | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 로딩 CScene, 스테이지 CScene, 클리어 CScene, 사망 CScene, CPlayer를 생성하고 초기화한다. |
| Update | int | 현재 CScene에 대한 Update함수를 호출한다. |
| LateUpdate | ini | 현재 CScene에 대한 LateUpdate함수를 호출한다. |

|  |  |  |
| --- | --- | --- |
| CScene | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | ItemList를 생성하고 ItemList와 CScene을 초기화한다. |
| Update | int | 아이템을 생성하고 오브젝트를 Update 해준다. |
| LateUpdate | int | 몬스터를 삭제하고 플레이어의 LateUpdate를 호출한다. |
| GetCurScene  MaxDistance | int | 스테이지 끝단의 거리를 가져온다. |
| UpdateMaxDistance | void | 현재 거리를 업데이트한다. |

|  |  |  |
| --- | --- | --- |
| CObjectManager | | |
| 함수 이름 | 리턴 타입 | 기능 |
| RegisterObject | void | 객체를 오브젝트 매니저에 등록한다. |
| GetObjectFromID | CObject\* | ID값을 받아 객체의 주소를 반환한다. |
| RemoveObject | void | 객체를 오브젝트 매니저에서 삭제한다. |

|  |  |  |
| --- | --- | --- |
| CObject | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 오브젝트의 이미지를 로드하고 오브젝트를 초기화해줍니다. |
| Update | void | 오브젝트를 업데이트합니다. |
| LateUpdate | void | 오브젝트를 후처리합니다. |
| Collision | bool | 오브젝트의 충돌을 확인합니다. |
| Changeimg |  |  |
| GetPos | POSITION | 오브젝트 이미지의 왼쪽 위 좌표를 리턴해줍니다. |
| GetVector | POSITION | 오브젝트의 진행 방향을 리턴해줍니다. |
| GetSize | \_SIZE | 오브젝트의 크기를 리턴해줍니다. |
| GetLife | bool | 오브젝트의 생존 여부를 리턴해줍니다. |
| GetHP | float | 오브젝트의 체력값을 리턴해줍니다. |
| SetPos | void | 오브젝트의 위치를 설정합니다. |
| SetVector | void | 오브젝트의 진행 방향을 설정합니다. |
| SetSize | void | 오브젝트의 사이즈를 설정합니다. |
| SetLife | void | 오브젝트의 생존 여부를 설정합니다. |
| SetHP | void | 오브젝트의 체력값을 설정합니다. |

|  |  |  |
| --- | --- | --- |
| CItem | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetEnbale | bool | 아이템 유효 여부를 확인합니다. |
| SetEnalbeFalse | void | 아이템을 비활성화합니다. |
| GetType | ITEM\_TYPE | 아이템 타입을 반환합니다. |

|  |  |  |
| --- | --- | --- |
| CBullet | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetEnbale | bool | 아이템 유효 여부를 확인합니다. |
| SetEnalbeFalse | void | 아이템을 비활성화합니다. |

|  |  |  |
| --- | --- | --- |
| CNetworkDevice | | |
| 함수 이름 | 리턴 타입 | 기능 |
| SendToNetwork | void | CMessageDispatcher로부터 메시지를 전달받아 가공해서 클라이언트로 전송한다. |
| RecvByNetwork | void | 클라이언트로부터 메시지를 받아 해석해서 CMessageDispatcher로 전달한다. |

|  |  |  |
| --- | --- | --- |
| CMessageDispatcher | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetMessages | Telegram\* | 메시지 큐의 데이터를 꺼내서 리턴해준다. |
| Discharge | void | 메시지를 특정 객체에게 전달해준다. |
| DispatchMessage | void | 메시지 큐에 메시지를 담는다. |

1. 클라이언트 모듈

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CCore | | | | |
| 함수 이름 | 리턴 타입 | | 기능 | |
| Init | bool | | CTimer, CSceneManager, CSoundManager, CObjectManage, CNetworkDevice를 초기화하고 서버로 메시지를 보냅니다. | |
| Input | void | | CSceneManager의 Input 함수를 호출한다. | |
| Update | int | | CSceneManger의 Update함수를 호출한다. | |
| LateUpdate | int | | CSceneManger의 LateUpdate함수를 호출한다. | |
| Logic | void | | 타이머를 업데이트하고 게임 루프를 실행하고 렌더링하고 서버로 메시지를 전송하고 다시 받는다. | |
| CTimer | | | | |
| 함수 이름 | | 리턴 타입 | | 기능 |
| Init | | bool | | 타이머를 초기화한다. |
| Update | | void | | 프레임 타임을 계산한다 |
| GetDeltaTime | | float | | 프레임 타임을 반환한다. |

|  |  |  |
| --- | --- | --- |
| CSceneManager | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 필요한 이미지들을 로드하고, 로딩 CScene, 스테이지 CScene, 클리어 CScene, 사망 CScene, CPlayer를 생성하고 초기화한다. |
| Input | void | 현재 CScene의 Input 함수를 호출합니다. |
| Update | int | 현재 CScene에 대한 Update함수를 호출한다. |
| LateUpdate | ini | 현재 CScene에 대한 LateUpdate함수를 호출한다. |
| Render | void | 현재 CScene의 Render 함수를 호출합니다. |

|  |  |  |
| --- | --- | --- |
| CScene | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 배경이미지를 로드하고 ItemList를 생성하고 ItemList와 CScene을 초기화한다. |
| imgLT\_Move\_Auto | void | 배경이미지를 스크롤링합니다. |
| Update | int | 아이템을 생성하고 오브젝트를 Update 해준다. |
| LateUpdate | int | 몬스터를 삭제하고 플레이어의 LateUpdate를 호출한다. |
| GetCurScene  MaxDistance | int | 스테이지 끝단의 거리를 가져온다. |
| Render | void | 렌더링을 한다. |
| UpdateMaxDistance | void | 현재 거리를 업데이트한다. |

|  |  |  |
| --- | --- | --- |
| CObjectManager | | |
| 함수 이름 | 리턴 타입 | 기능 |
| RegisterObject | void | 객체를 오브젝트 매니저에 등록한다. |
| GetObjectFromID | CObject\* | ID값을 받아 객체의 주소를 반환한다. |
| RemoveObject | void | 객체를 오브젝트 매니저에서 삭제한다. |

|  |  |  |
| --- | --- | --- |
| CObject | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Init | bool | 오브젝트의 이미지를 로드하고 오브젝트를 초기화해줍니다. |
| Update | void | 오브젝트를 업데이트합니다. |
| LateUpdate | void | 오브젝트를 후처리합니다. |
| Collision | bool | 오브젝트의 충돌을 확인합니다. |
| Changeimg |  |  |
| GetPos | POSITION | 오브젝트 이미지의 왼쪽 위 좌표를 리턴해줍니다. |
| GetVector | POSITION | 오브젝트의 진행 방향을 리턴해줍니다. |
| GetSize | \_SIZE | 오브젝트의 크기를 리턴해줍니다. |
| GetLife | bool | 오브젝트의 생존 여부를 리턴해줍니다. |
| GetHP | float | 오브젝트의 체력값을 리턴해줍니다. |
| SetPos | void | 오브젝트의 위치를 설정합니다. |
| SetVector | void | 오브젝트의 진행 방향을 설정합니다. |
| SetSize | void | 오브젝트의 사이즈를 설정합니다. |
| SetLife | void | 오브젝트의 생존 여부를 설정합니다. |
| SetHP | void | 오브젝트의 체력값을 설정합니다. |
| HandleMessage | void | 메시지를 처리합니다. |

|  |  |  |
| --- | --- | --- |
| CPlayer | | |
| 함수 이름 | 리턴 타입 | 기능 |
| SetHitTractorBeam | void | m\_bHit\_TractorBeam(트랙터 공격 피격 여부)를 설정합니다. |
| GetHitTractorBeam | void | m\_bHit\_TractorBeam(트랙터 공격 피격 여부)를 리턴해줍니다. |
| GetmyBulletList | CBulletList\* | 플레이어 총알 리스트를 리턴해줍니다. |
| IsPlayerDie | bool | 플레이어 사망여부를 리턴해줍니다. |
| AddBullet | void | 총알의 숫자를 확인하는 변수를 +1 해줍니다. |
| AddAttack | void | 플레이어의 공격력을 증가시키기위한 함수입니다. |
| SetINVINVIBILITY | void | 플레이어의 무적상태를 표현하는 m\_bINVINVIBILITY를 설정합니다. |
| GetINVINVIBILITY | bool | 플레이어의 무적상태를 표현하는 m\_bINVINVIBILITY를 리턴합니다. |
| SetHPPrototype | void |  |
| GetSpeed | float | 플레이어 속도를 리턴합니다. |
| SetSpeed | void | 플레이어 속도를 설정합니다. |
| GetMoveDistance | float | 플레이어의 이동 거리를 계산해 리턴합니다. |
| SetType | void | 플레이어의 Type을 설정해줍니다. |
| GetMyType | OBJECT\_TYPE | 플레이어의 Type을 리턴합니다. |
| RenderBulletList | void | 플레이어 총알을 모두 렌더링합니다. |

|  |  |  |
| --- | --- | --- |
| CMonster | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetIsDie | bool | Monster생존 여부를 확인합니다. |
| RenderExplode | void | 폭발 애니메이션을 렌더링합니다. |
| Run | void | CMonster의 Update, LateUpdate, Render를 호출합니다. |
| GetState | MONSTER\_STATE | 몬스터의 상태를 리턴합니다. |

|  |  |  |
| --- | --- | --- |
| CItem | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetEnbale | bool | 아이템 유효 여부를 확인합니다. |
| SetEnalbeFalse | void | 아이템을 비활성화합니다. |
| GetType | ITEM\_TYPE | 아이템 타입을 반환합니다. |

|  |  |  |
| --- | --- | --- |
| CBullet | | |
| 함수 이름 | 리턴 타입 | 기능 |
| GetEnbale | bool | 아이템 유효 여부를 확인합니다. |
| SetEnalbeFalse | void | 아이템을 비활성화합니다. |

|  |  |  |
| --- | --- | --- |
| CSoundManager | | |
| 함수 이름 | 리턴 타입 | 기능 |
| playSound | bool | 배경음악을 재생합니다. |
| stopSound | bool | 배경음악 재생을 중지합니다. |
| PlaySound\_Bossbgm | bool | 보스 배경음악을 재생합니다. |
| playSound\_Effect | bool | 효과음을 재생합니다. |
| playSound\_Effect\_Explode | bool | 폭발 효과음을 재생합니다. |
| HandleMessage | void | 메시지를 처리합니다. |

|  |  |  |
| --- | --- | --- |
| CNetworkDevice | | |
| 함수 이름 | 리턴 타입 | 기능 |
| SendToNetwork | void | Message들을 형식을 변환해 서버로 보냅니다. |
| RecvByNetwork | void | 서버로부터 메시지를 받아 해석합니다. |
| AddMessage | void | 메시지 큐에 메시지를 추가합니다. |

|  |  |  |
| --- | --- | --- |
| CMessageDispatcher | | |
| 함수 이름 | 리턴 타입 | 기능 |
| Discharge | void | 메시지를 특정 객체에게 전달해준다. |
| DispatchMessage | void | 메시지 큐에 메시지를 담는다. |

|  |  |  |  |
| --- | --- | --- | --- |
| Msg\_type | sender | receiver | extrainfo |
| msg\_objectCreate | sever/client | objectID | ObjectType, position |
| msg\_objectMove | server/client | objectID | position |
| msg\_playerHit | server | objectID | X |
| msg\_objectChangeState | server | objectID | ObjectState |
| msg\_playSound | server | objectID | int |
| msg\_changeScene | server | objectID | int |
| msg\_clientReady | client | objectID | X |

1. 메시지 구조체
2. 패킷 구조 (고정 + 가변 길이 데이터 전송 방식)

메시지 타입 별 개수(28 바이트) + 메시지 데이터(n바이트)

메시지 데이터의 크기 n = (msg\_objectCreate의 개수) \* 24 +

(msg\_objectMove의 개수) \* 20 +

(msg\_playerHit의 개수) \* 4 + (msg\_objectChangeState의 개수) \* 8 + (msg\_playSound의 개수) \* 8 + (msg\_changeScene의 개수) \* 8 + (msg\_clientReady의 개수) \* 4

1. 패킷 송수신 방식

CNetworkDevice는 각 메시지 타입별로 Telegram 배열을 지닌다.

CNetworkDevice::AddMessage() 를 호출하여 네트워크 디바이스에 메시지를 전달하여 저장한다.

CNetworkDevice::SendToNetwork() 가 호출되면 네트워크 디바이스는 먼저 각 메시지 타입 별 메시지의 개수를 int형 배열에 저장한다.

메시지의 종류는 서버와 클라이언트가 모두 알고 있으므로 고정길이 패킷으로 전달이 가능하므로 이렇게 만든 int형 배열을 고정길이 패킷으로 먼저 송신한다.

그리고 각 메시지 타입별로 나뉘어서 저장된 배열에서 Telegram을 하나씩 꺼내

[receiver, extrainfo] 순으로 직렬화하여 가변길이 패킷 버퍼에 순차적으로 넣는다.

이때, 메시지타입의 값이 낮은 순부터 넣는다.

반대로 CNetworkDevice::RecvByNetwork() 가 호출되면 받아들인 고정길이 패킷을 int형 배열로 받아 각 인덱스를 순차적으로 접근하여 메시지 타입별 메시지 개수를 저장한다.

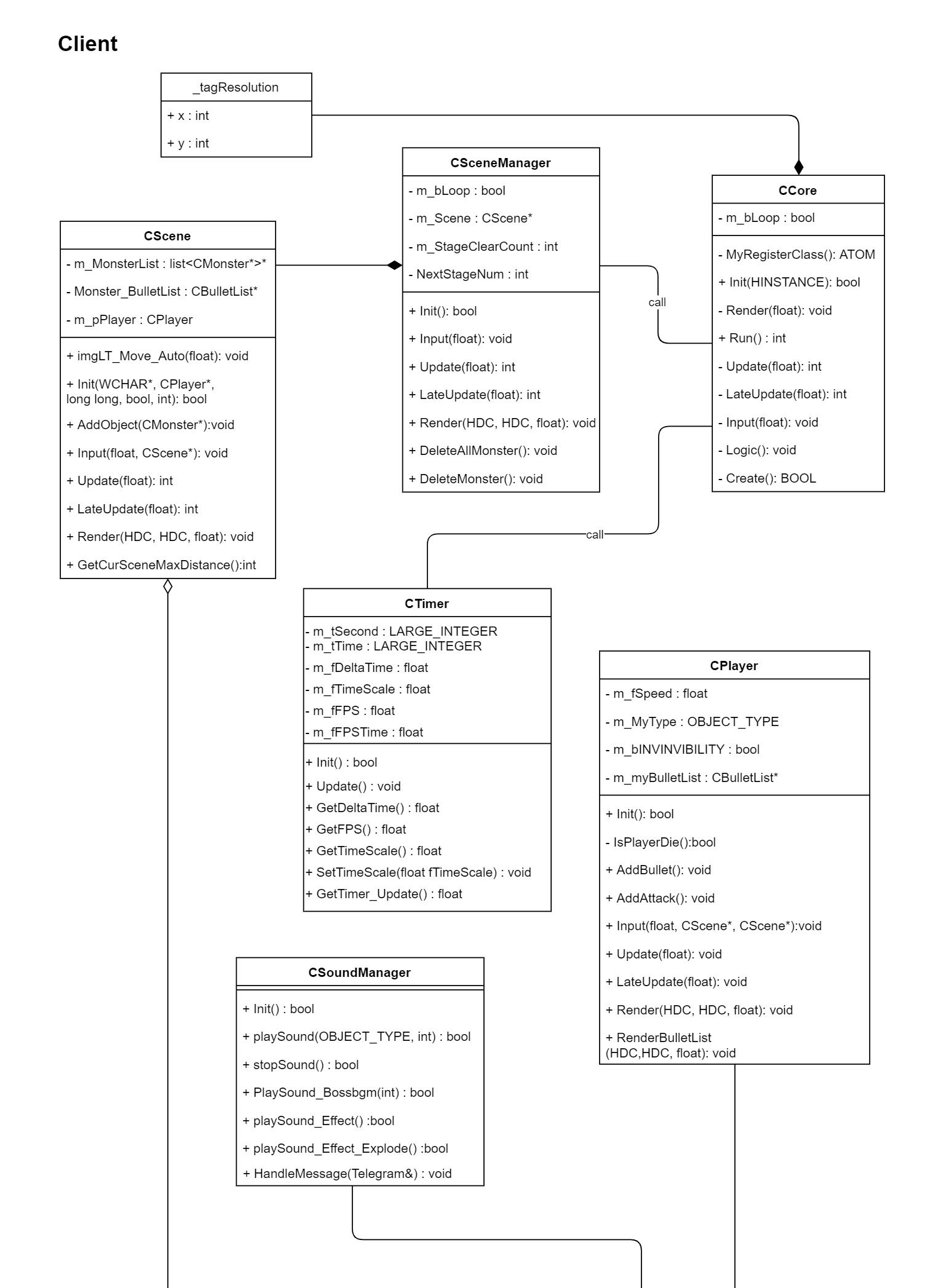
이후 가변길이 패킷을 받아 이중 반복문을 돌린다.

첫번째 반복은 int형 배열의 크기만큼 반복하며 두번째 반복은 int형 배열의 현재 i 번재 원소의 값만큼 반복한다.

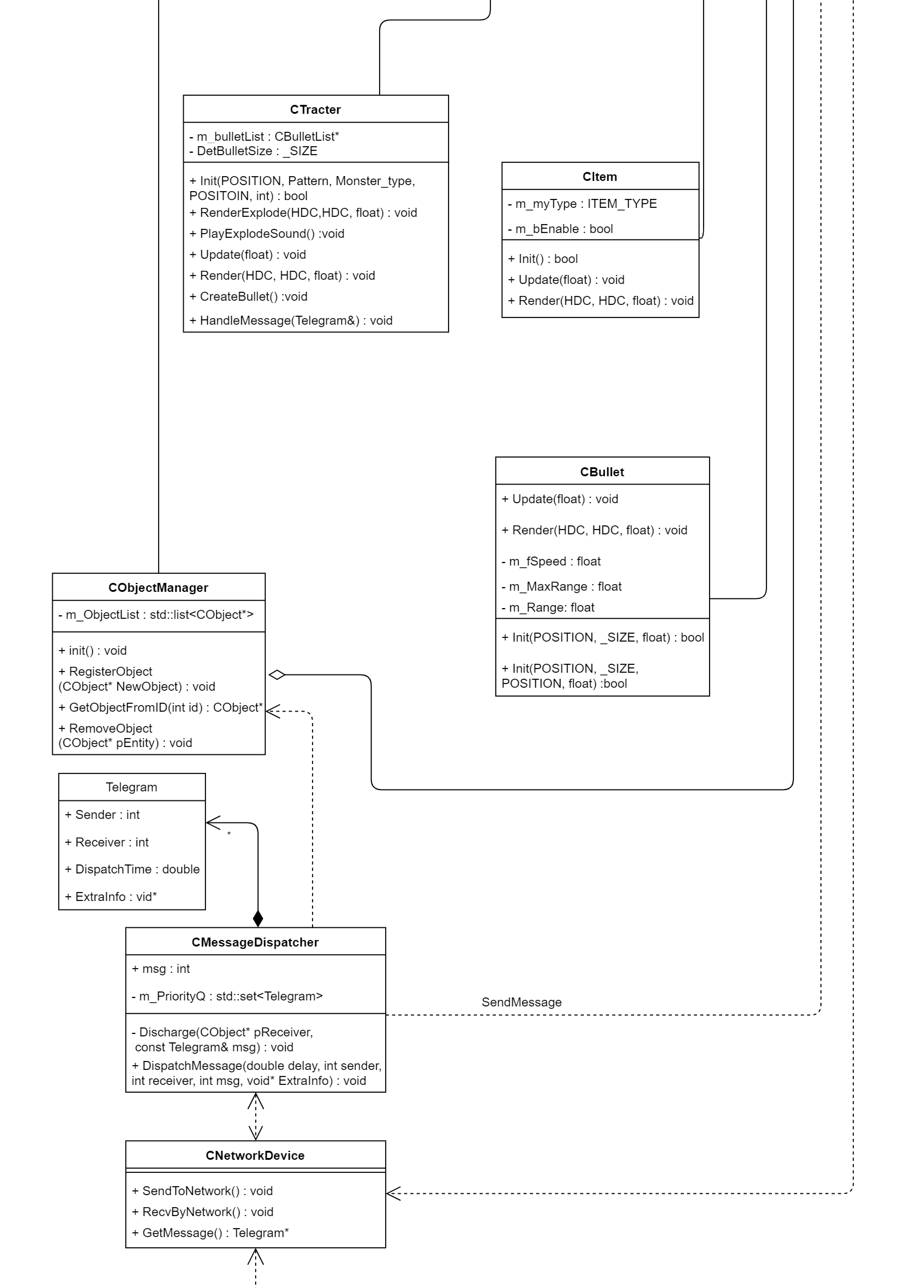
i번째 원소에는 메시지타입별 메시지의 개수가 저장되어 있다.

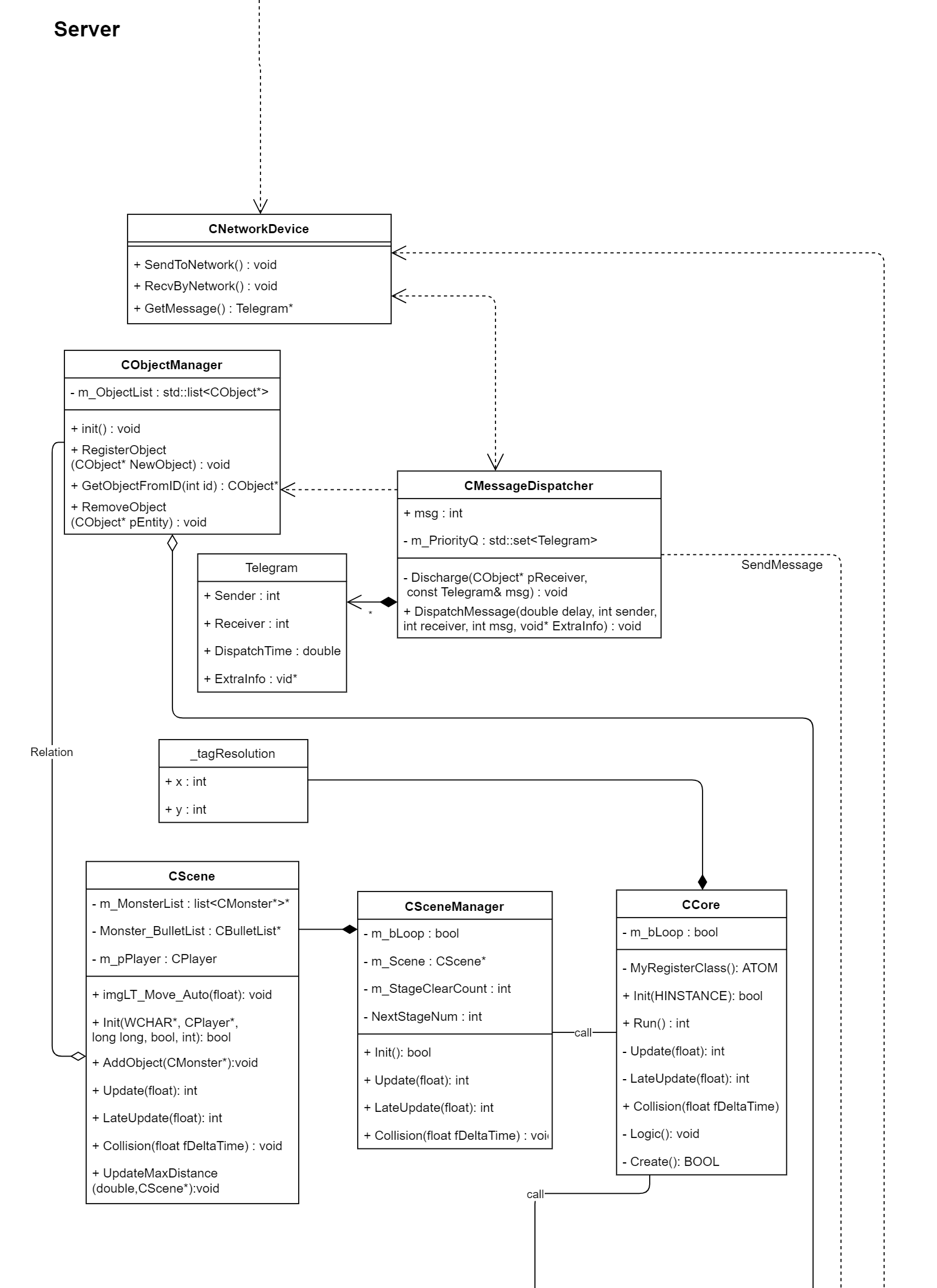
이 루프를 반복하면서 각각의 메시지 타입 별로 미리 계산해 놓은 메시지별 바이트 수만큼 패킷을 읽어 Telegram객체를 만들어 저장한다.

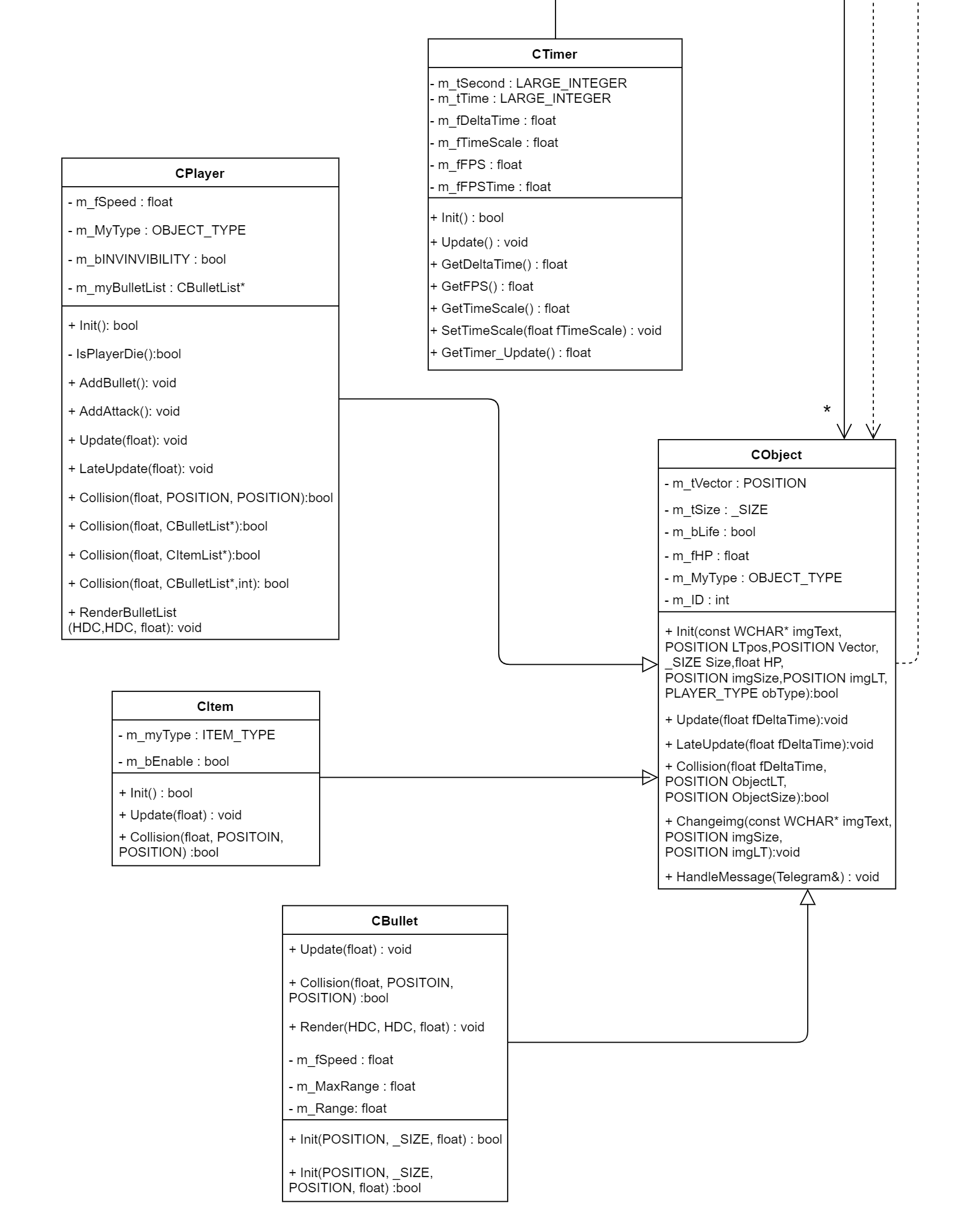
이후, CMessageDispatcher에서 CNetworkDevice로부터 메시지를 넘겨받아 게임 내 객체에게 메시지를 전달한다.

1. UML Class Diagram









텍스트, 영수증, 스크린샷이(가) 표시된 사진

자동 생성된 설명

4. 역할분담

|  |  |  |  |
| --- | --- | --- | --- |
| 구현내용 | 이지형 | 최해성 | 홍명진 |
| 서버 프레임워크 구현 | ✓ | ✓ | ✓ |
| 씬 전환 | ✓ |  |  |
| 몬스터 생성 및 시뮬레이션 | ✓ |  |  |
| 아이템 생성 및 시뮬레이션 |  | ✓ |  |
| 총알 생성 및 시뮬레이션 |  |  | ✓ |
| 충돌처리 | ✓ | ✓ |  |
| CObjectManager 구현 |  | ✓ |  |
| CObject 구조 및 상속 구조 변경 |  |  | ✓ |
| CMessageDispatcher 구현 | ✓ | ✓ | ✓ |
| CNetworkDevice 구현 | ✓ | ✓ | ✓ |
| 플레이어 사망 및 게임 종료 구현 |  | ✓ |  |

1. 서버

|  |  |  |  |
| --- | --- | --- | --- |
| 구현내용 | 이지형 | 최해성 | 홍명진 |
| 프레임워크 수정 | ✓ | ✓ | ✓ |
| 플레이어 입력 처리 |  |  | ✓ |
| CObjectManager 구현 |  | ✓ |  |
| CObject 구조 및 상속 구조 변경 |  |  | ✓ |
| CMessageDispatcher 구현 | ✓ | ✓ | ✓ |
| CNetworkDevice 구현 | ✓ | ✓ | ✓ |
| 사운드 재생 방식 변경 | ✓ |  |  |

1. 클라이언트

5. 개발환경

* 개발도구: VisualStudio
* 운영체제: Windows 10 x86
* 통신 프로토콜: TCP/IP
* 외부 라이브러리: FMOD
* 관리 툴: GitHub

|  |  |  |  |
| --- | --- | --- | --- |
| 날짜 | 이지형 | 최해성 | 홍명진 |
| 11/5 | 기존 코드 분석 | 기존 코드 분석 | 기존 코드 분석 |
| 11/6 | 기존 코드 분석 | 기존 코드 분석 | 기존 코드 분석 |
| 11/7 |  |  |  |
| 11/8 | 서버 CCore 구현 | 클라 CCore 구현 | CObject 상속 구조 변경 |
| 11/9 | RecvByNetwork() | CObjectManager 구현 | SendToNetwork() |
| 11/10 |  |  |  |
| 11/11 | Discharge() | DispatchMessage() | AddMessage() |
| 11/12 | Monster::HandleMessage() | SceneManager::HandleMessage() | Player::HandleMessage() |
| 11/13 |  |  |  |
| 11/14 | Tracter::HandleMessage()  Boss::HandleMessage() | Item::HandleMessage()  SoundManager::HandleMessage() | Bullet::HandleMessage()  서버 내부 테스트 |
| 11/15 | ~~Boss::HandleMessage()~~  CScene::HandleMessage()  msg\_ClientReady 추가 | ~~SoundManager::HandleMessage()~~  Client에서 테스트 메시지 보내고 받게하기 | ~~서버 내부 테스트~~  Client메인화면 렌더링하게하기 |
| 11/16 | 서버&클라이언트 외부 테스트 | 클라이언트 내부 테스트 | 서버 CSceneManager |
| 11/17 |  |  |  |
| 11/18 |  |  |  |
| 11/19 | ~~서버 CObject 구현~~ | 서버 CScene 구현 | 서버 CBullet 구현 |
| 11/20 | ~~서버 CTimer 구현~~ | 서버 CItem 구현  ItemList를 STL로 전환 | ~~BulletList를 STL로 전환~~ |
| 11/21 | ~~씬 전환 수정~~  CardinalSpline() | 서버 CItem 버그 수정  클라이언트한테 Item 띄우기 | Monster Bullet 구현 |
| 11/22 | ~~실행 테스트~~  MonsterPattern 1,2 구현  MonsterPattern 1,2 구현  MonsterPattern 5 구현 | 서버 몬스터 구현  ObjectID 동기화  ~~실행 테스트~~ | ~~실행 테스트~~  서버 CBullet 구현  Monster Bullet 구현 |
| 11/23 | 작업물 Merge 후 실행테스트 | 임계영역 오류 해결  작업물 Merge 후 실행테스트 | 작업물 Merge 후 실행테스트 |
| 11/24 | ~~MonsterPattern 1,2 구현~~ | ~~ItemList를 STL로 전환~~ | Player 입력처리 |
| 11/25 | ~~MonsterPattern 3,4 구현~~ | CScene Item 구현 | Player Bullet 구현 |
| 11/26 | ~~MonsterPattern 5,6 구현~~ |  |  |
| 11/27 |  | 플레이어 & 총알 충돌 | Boss Pattern 1, 2 구현 |
| 11/28 | 아이템 & 플레이어 충돌 | 플레이어 사망 구현 | Boss Pattern 3, 4 구현 |
| 11/29 | 몬스터 & 총알 충돌 | 서버 GameOver Scene 구현 | Boss Pattern 5 구현 |
| 11/30 | 몬스터 삭제 알고리즘 | 클라이언트 GameOver Scene 구현 |  |
| 12/01 | CSoundManager 수정 |  |  |
| 12/02 |  |  |  |
| 12/03 | 버그 수정 및 동기화 | 버그 수정 및 동기화 | 버그 수정 및 동기화 |
| 12/04 | 버그 수정 및 동기화 | 버그 수정 및 동기화 | 버그 수정 및 동기화 |
| 12/05 | 버그 수정 및 동기화 | 버그 수정 및 동기화 | 버그 수정 및 동기화 |
| 12/06 | 버그 수정 및 동기화 | 버그 수정 및 동기화 | 버그 수정 및 동기화 |
| 12/07 | 버그 수정 및 동기화 | 버그 수정 및 동기화 | 버그 수정 및 동기화 |

6. 일정