**네트워크 게임 프로그래밍 추진 계획서**

|  |  |
| --- | --- |
| 학번 | 이름 |
| 2013182001 | 강수빈 |
| 2013182010 | 김병진 |
| 2013182030 | 이상기 |

목차

내용

[애플리케이션 기획 3](#_Toc497956189)

[개요 3](#_Toc497956190)

[네트워크 기능 6](#_Toc497956191)

[네트워크 구조 개요도 6](#_Toc497956192)

[서버 7](#_Toc497956193)

[클라이언트 7](#_Toc497956194)

[High-Level Design 9](#_Toc497956195)

[Low-Level Design 13](#_Toc497956196)

[팀원 별 역할 분담 14](#_Toc497956197)

[개발 환경 15](#_Toc497956198)

[테스트 환경 15](#_Toc497956199)

[개발 일정 16](#_Toc497956200)

[11/9~11/15 17](#_Toc497956201)

[11/16~11/22 18](#_Toc497956202)

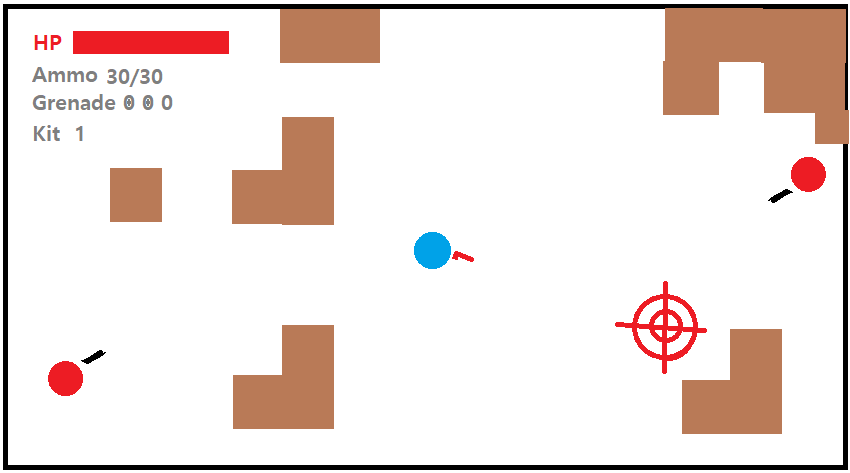
[11/23~11/29 19](#_Toc497956203)

[11/30~12/6 20](#_Toc497956204)

[12/7~11/13 21](#_Toc497956205)

# 애플리케이션 기획

### 개요



제목 :: Network Shooting Game Simulator

개요 :: 로비에서 네트워크 접속을 통해 방에 접속하여 플레이하는 2D 탑 뷰 슈팅게임이다.

이 게임은 키보드 WASD 키로 움직이며 플레이어가 마우스를 사용하여 적을 조준하여 사격하며 진행하는 게임으로서 게임 시작 이후 해당 룸에 플레이어가 단 한 명만 남게 되면 게임이 종료된다.

조작법

|  |  |
| --- | --- |
| 마우스 이동 | 조준점 이동 |
| 마우스 왼쪽 클릭 | 사격 |
| W | Y축 양의 방향 ( 위 방향 ) 으로 이동 |
| S | Y축 음의 방향 ( 아래 방향 ) 으로 이동 |
| A | X축 음의 방향 ( 왼쪽 방향 ) 으로 이동 |
| D | X축 양의 방향 ( 오른쪽 방향 ) 으로 이동 |
| G | 수류탄 투척 |
| R | 재장전 |

* 플레이어

플레이어는 서버에 접속한 클라이언트가 조작하는 에이전트다.

플레이어는 게임 시작 시 서버에서 정의한 좌표에 생성된다.

플레이어는 30발 탄알집을 사용해서 사격을 할 수 있다.

사격을 한번 시행할 시마다 탄알집에서 탄환을 하나 소비한다.

탄알집의 모든 탄환을 소비하면 재장전 커맨드를 입력하여 탄환을 보충할 수 있다.

각 플레이어는 3발의 수류탄, 그리고 하나의 터렛 설치 키트 하나를 지급받는다.

플레이어는 클라이언트에서 키보드와 마우스를 사용하여 에이전트를 조작할 수 있다.

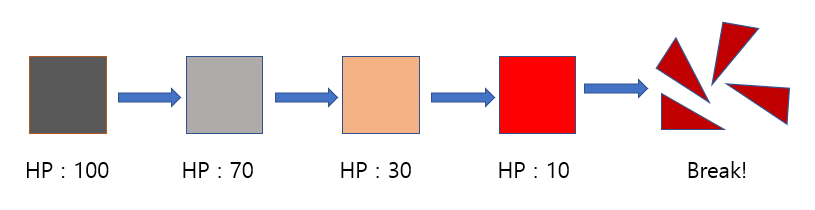
플레이어는 G키를 입력하여 바라보는 방향으로 수류탄을 투척할 수 있다.

수류탄은 투척된 직후 3초 이후 터지며 수류탄 반경 내에 있는 모든 오브젝트에게 손상을 준다.

플레이어는 터렛 설치 키트를 사용해 플레이어의 터렛을 설치할 수 있다.

플레이어의 체력이 0이 되면 사망한다.

* 벽



벽은 서버 호스트에 저장되어 있는 맵 데이터를 읽어 데이터에 따라 벽 오브젝트를 배치한다.

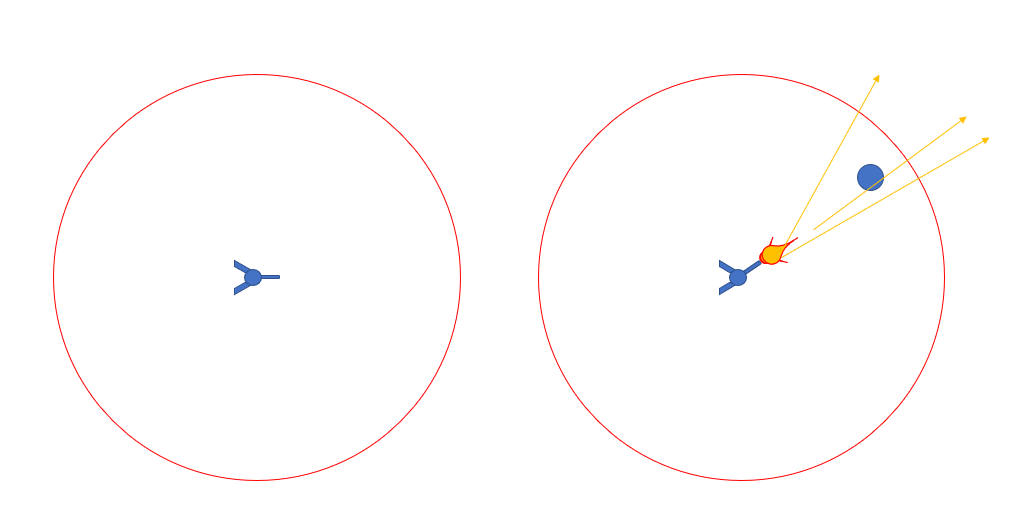
벽은 플레이어와 투사체와 충돌판정을 하며 플레이어가 이동하는 것을 막는 장애물의 역할과 투사체의 진행을 가로막는 엄폐물의 역할을 한다.

벽은 고유한 체력을 가지고 있다. 체력이 줄어들면 줄어들수록 손상되는 이미지를 출력하며 벽이 가진 체력 수치가 0이 되면 파괴된다.

* 에이전트

AI는 플레이어가 따로 컨트롤하지 않는 에이전트로서 서버에서 미리 정의된 상태기계에 따라 움직이거나 사격하는 행위를 한다.

1. 터렛



* 터렛의 사격 도식 :: 터렛은 고유의 사거리를 지니고, 사격을 할 수 있다.

터렛은 맵 상에 무작위로 배치되거나, 플레이어가 아이템을 사용해서 설치했을 때 생성되는 에이전트이다.

다른 플레이어가 설치한 터렛의 경우 자신이 어느 플레이어에게 설치되었는지를 기억한다.

터렛은 자신이 설치한 플레이어를 제외한 모든 에이전트를 적으로 간주한다.

터렛은 각자가 고유의 사정거리를 지니고 있다. 사거리 내부에 적이 들어오면 해당 적의 방향으로 사격을 개시한다.

1. 자동으로 움직이는 에이전트 ( 이하 AI로 지칭한다. )

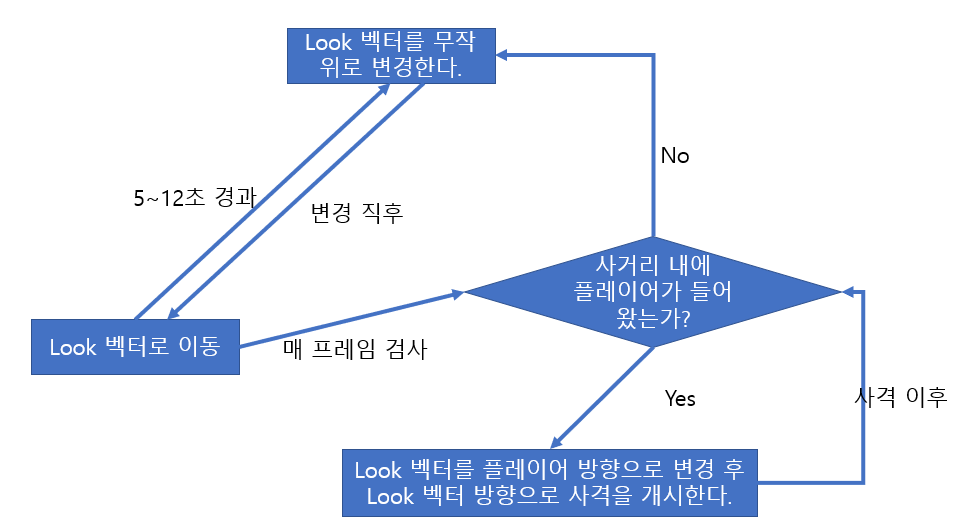
AI는 5 + rand()%7초 동안 바라보고 있는 방향으로 이동한다.

AI 는 이동 상태에서 5 + rand()%7초가 지나면 무작위로 바라보는 방향을 변경한다.

AI는 매 프레임마다 AI의 사거리 내에 다른 플레이어가 있는지 판정을 한다.

플레이어가 사거리 내에 있을 시 플레이어 방향으로 사격을 개시한다.

사격을 행하는 것은 플레이어가 AI의 사거리 밖으로 벗어날 때 까지 진행한다.



* AI의 유한상태기계

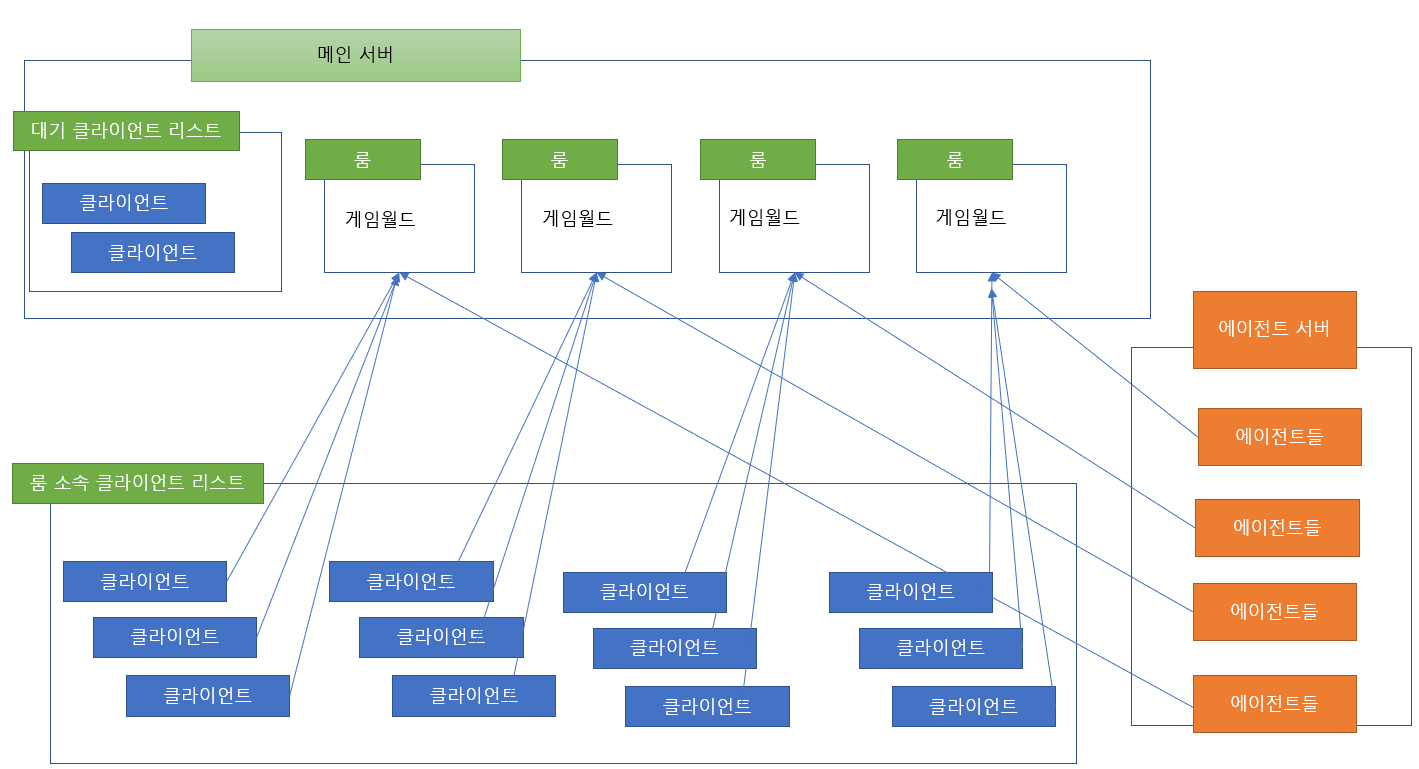
## 네트워크 기능

클라이언트와 서버간 데이터들은 반드시 다음 사항을 보장한다.

네트워크 통신을 통해 주고 받는 데이터들은 유실이 없어야 한다.

따라서 서버와 클라이언트 간에 모든 데이터 통신은 **TCP 프로토콜**을 사용한다.

### 네트워크 구조 개요도



### 서버

#### 메인 서버

메인 서버는 게임에 접속한 클라이언트들이 룸에 접속하여 게임을 진행할 때 발생하는 모든 상호작용을 컨트롤하고 상호작용의 결과들을 클라이언트들에게 다시 송신하는 역할을 한다.

###### 룸의 리스트

룸은 게임이 진행되고 있는 월드를 지칭한다.

생성된 룸은 서버에서 연산한 정보들을 룸에 소속된 클라이언트 들에게 데이터를 보내기 위해 해당 룸에 소속된 클라이언트들의 정보를 가진다.

###### 클라이언트 리스트

클라이언트 리스트는 메인 서버에 접속한 클라이언트들의 정보를 저장하는 자료구조다. 클라이언트 리스트는 대기 클라이언트 리스트와 룸 소속 클라이언트 리스트로 나뉜다.

대기 클라이언트 리스트

대기 클라이언트 리스트는 서버에 접속한 후 아직 게임을 진행하지 않고 대기하는 클라이언트들을 관리한다.

룸 소속 클라이언트 리스트

하나의 룸에 소속되어 게임을 진행하고 있는 클라이언트들을 관리한다.

메인 서버는 클라이언트 리스트와 생성된 룸의 리스트들을 관리한다.

#### 에이전트 서버

에이전트 서버는 메인 서버에서 게임이 진행될 때 메인 서버의 룸에서 행동하는 에이전트들의 상태기계와 상태의 전환들을 관리하는 역할을 한다.

###### 룸의 리스트

룸의 리스트는 메인 서버가 가지고 있는 룸의 리스트와 동기화 한다.

### 클라이언트

클라이언트는 서버가 생성한 룸에서 다음과 같은 정보를 가진다.

클라이언트 캐릭터의 체력, 탄약, 수류탄, 터렛 킷 개수

다른 에이전트 오브젝트의 정보

클라이언트에서는 메인 서버가 보낸 월드 정보를 수신을 받고 그 정보를 바탕으로 게임 화면을 렌더링한다.

##### 클라이언트의 메인 서버로의 접속

클라이언트는 프로그램을 실행한 후 메인 서버에게 접속을 요청한다.

클라이언트가 서버에 접속을 성공하면 게임을 진행할 룸이 생성될 때까지 대기한다.

메인 서버가 룸을 생성하면 대기하던 클라이언트들은 생성된 룸으로 들어가 게임플레이를 진행한다.

게임이 끝나면 다시 대기한다.

##### 메시지 리시버

메시지 리시버는 클라이언트와 메인 서버, 그리고 에이전트 서버가 공통적으로 사용하는 스레드 함수이다.

메시지 리시버에서는 호스트에게 전송된 메시지들을 수신하는 역할을 하며, 수신된 모든 메시지를 메시지 큐에 저장하는 역할을 수행한다.

##### 클라이언트와 메인 서버간 네트워크 통신을 위한 메시지 큐

메시지 큐는 메시지 리시버가 받아온 메시지들을 저장하는 자료구조다.

##### 액션 메시지

액션 메시지는 클라이언트 또는 에이전트가 메인 서버에게 보내는 메시지를 지칭한다.

액션 메시지는 클라이언트 또는 에이전트의 변경할 행동을 나타내는 메시지다.

클라이언트는 조작 커맨드를 입력할 시 서버에게 액션 메시지를 송신한다.

에이전트의 경우 상태 전환이 발생할 시 서버에게 액션 메시지를 송신한다.

액션 메시지는 이동, 사격, 터렛 설치, 재장전 타입을 가지고 행동을 갱신할 오브젝트의 ID와 변경 정보를 가진다.

그 다음 클라이언트는 응답이 올 때까지 가장 최근에 했던 행동을 유지한다.

서버는 클라이언트와 에이전트의 액션 메시지를 받아 메시지 큐에 추가한다.

##### 허용 가능한 메시지

예를 들어 이미 클라이언트가 사망하였는데 사망한 플레이어가 보낸 액션 메시지가 메인 서버의 메시지 큐에 남아 있을 수 있다.

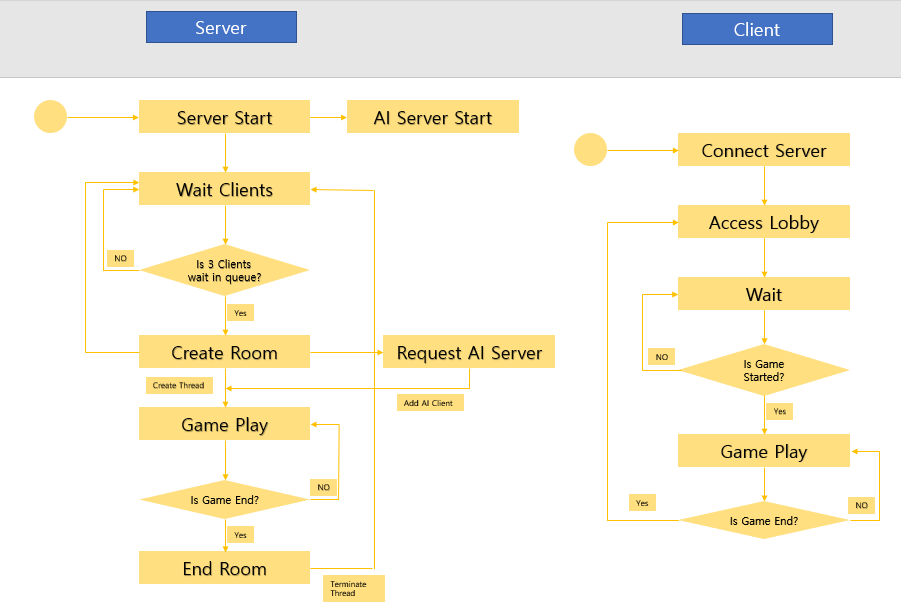
이를 위해 메시지 큐에서 메시지를 처리하기 전에 해당 메시지가 허용 가능한 메시지인지에 대한 판별을 우선적으로 시행한다.

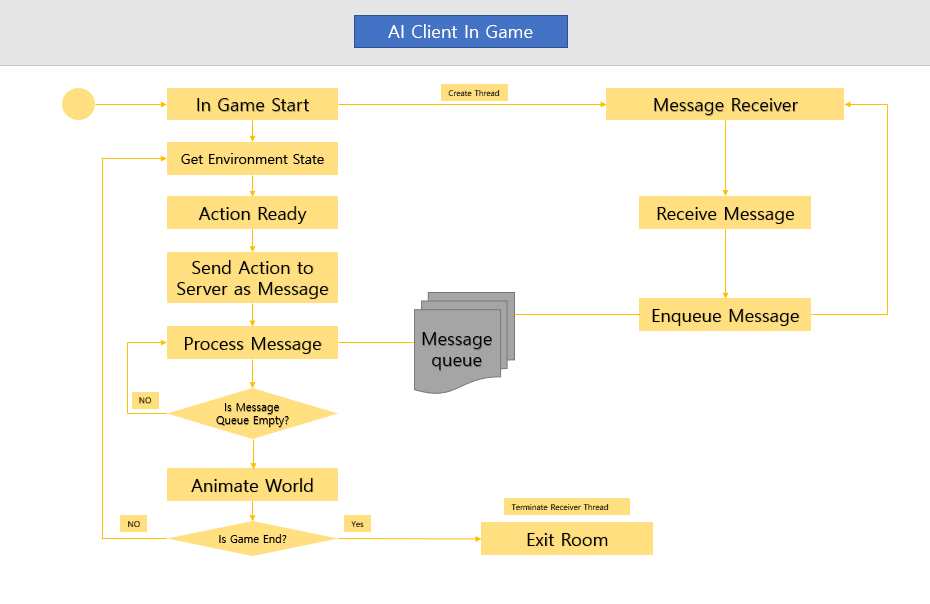
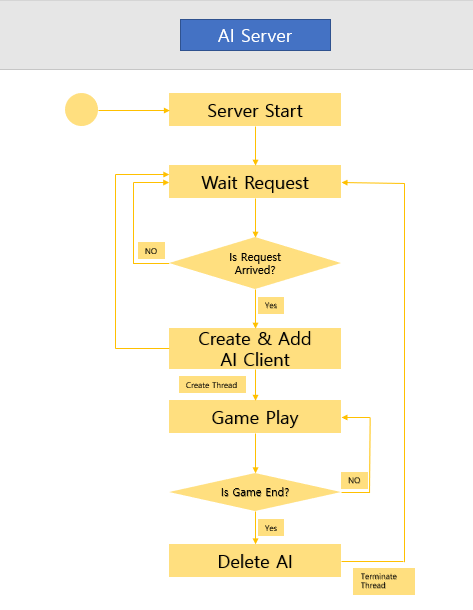
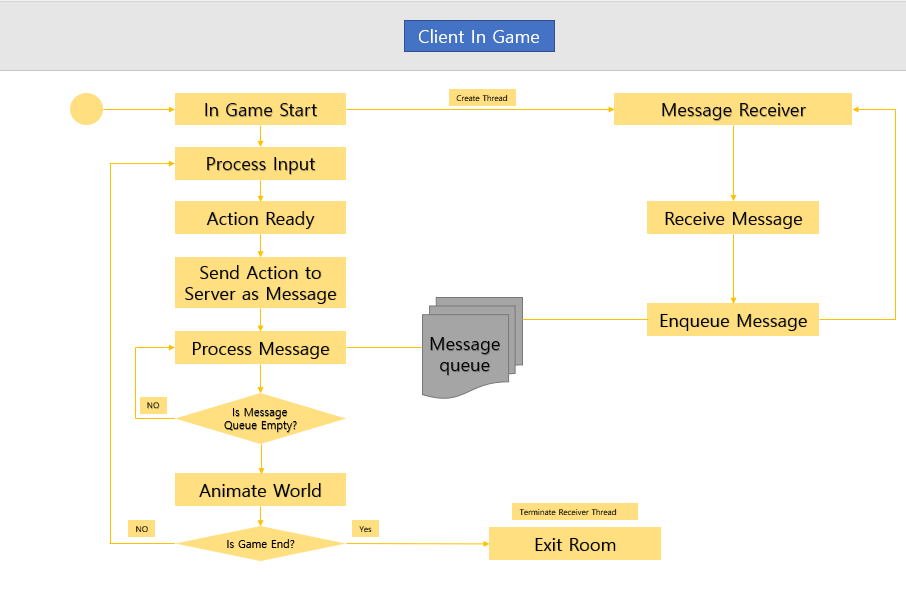
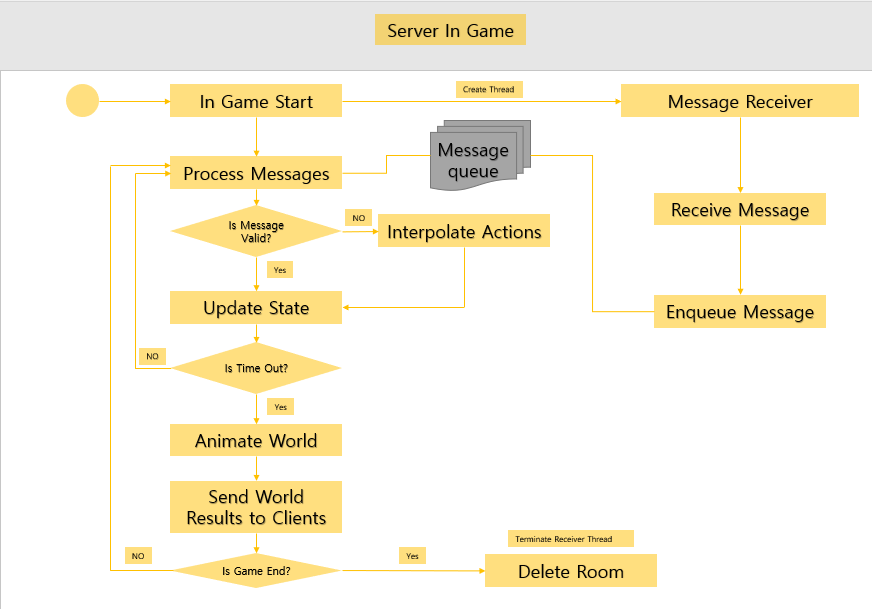
만약 해당 메시지가 허용 가능한 메시지가 아닐 경우 해당 메시지를 처리하지 않는다.

##### 메인 서버가 클라이언트와 에이전트에게 보내는 메시지

메인 서버는 메인 서버가 처리한 월드의 결과값들을 클라이언트와 에이전트들에게 송신한다.

# High-Level Design





# Low-Level Design

# 팀원 별 역할 분담

# 개발 환경

|  |  |
| --- | --- |
| 개발 PC 운영체제 | Microsoft Windows 10 |
| 개발 컴파일러 | Visual Studio 2017 |
| 개발 언어 | C / C++ |
| 개발 그래픽 라이브러리 | Direct2D |

## 테스트 환경

|  |  |
| --- | --- |
| 테스트 PC 운영체제 | Microsoft Windows 10 |
|  |  |
|  |  |
|  |  |

## 개발 일정

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 작업내용 | 시작일 | 종료예정일 | 상태 | 11 | | | | | | | | | | | | | | | | | | | | | | 12 | | | | | | | | | | | | |
| 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 게임 어플리케이션 개발 | 11/9 | 11/15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 네트워크 통신을 위한 메시지 큐 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 서버의 룸 생성 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 클라이언트와 메인 서버간 룸 접속 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 메인 서버와 클라이언트간 액션 메시지 송수신 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 에이전트 상태기계 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 메인 서버 룸 생성시 에이전트 할당 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 에이전트 서버와 메인 서버간 상호작용 구현 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

## 11/9~11/15

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11/9~11/15 | | 11/9 | 11/10 | 11/11 | 11/12 | 11/13 | 11/14 | 11/15 |
| 강수빈 | |  |  |  |  |  |  |  |
| 김병진 | |  |  |  |  |  |  |  |
| 이상기 | |  |  |  |  |  |  |  |
| 11/9 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/10 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 게임 어플리케이션 프레임워크 및 필요한 오브젝트 클래스 작성 | | | | | | |
| 11/11 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 플레이어 input processing 부분 작성 및 오브젝트 상호작용 구현 | | | | | | |
| 11/12 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | AI와 리소스를 제외한 싱글 플레이 상태의 게임 플레이 구현 | | | | | | |
| 11/13 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메시지 큐 제작 | | | | | | |
| 11/14 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/15 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |

## 11/16~11/22

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11/16~11/22 | | 11/16 | 11/17 | 11/18 | 11/19 | 11/20 | 11/21 | 11/22 |
| 강수빈 | |  |  |  |  |  |  |  |
| 김병진 | |  |  |  |  |  |  |  |
| 이상기 | |  |  |  |  |  |  |  |
| 11/16 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/17 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 클라이언트 통신 오브젝트 구현 | | | | | | |
| 11/18 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메시지 리시버 구현 및 메인 서버와 연결 | | | | | | |
| 11/19 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 룸의 게임 월드 실행 스레드 함수 구현 | | | | | | |
| 11/20 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 클라이언트에 메시지 큐 적용 | | | | | | |
| 11/21 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/22 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |

## 11/23~11/29

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11/23~11/29 | | 11/23 | 11/24 | 11/25 | 11/26 | 11/27 | 11/28 | 11/29 |
| 강수빈 | |  |  |  |  |  |  |  |
| 김병진 | |  |  |  |  |  |  |  |
| 이상기 | |  |  |  |  |  |  |  |
| 11/23 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/24 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메인 서버와의 메시지 송수신 테스트 | | | | | | |
| 11/25 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메시지 큐 처리 테스트 | | | | | | |
| 11/26 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메시지 큐 스레드 동기화 완료 | | | | | | |
| 11/27 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 여러 개의 클라이언트 구동 테스트 | | | | | | |
| 11/28 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 11/29 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |

## 11/30~12/6

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11/30~12/6 | | 11/30 | 12/1 | 11/2 | 11/3 | 11/4 | 11/5 | 11/6 |
| 강수빈 | |  |  |  |  |  |  |  |
| 김병진 | |  |  |  |  |  |  |  |
| 이상기 | |  |  |  |  |  |  |  |
| 11/30 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 12/1 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | AI 에이전트 오브젝트를 클라이언트에 추가 | | | | | | |
| 12/2 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 에이전트를 포함한 모든 오브젝트 상태 갱신 테스트 | | | | | | |
| 12/3 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 여러 개의 클라이언트 구동으로 메인 서버 룸 리스트 테스트 | | | | | | |
| 12/4 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 메인서버-클라이언트의 로컬통신 작동 테스트 | | | | | | |
| 12/5 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 12/6 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |

## 12/7~11/13

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 12/7~12/13 | | 12/7 | 12/8 | 12/9 | 12/10 | 12/11 | 12/12 | 12/13 |
| 강수빈 | |  |  |  |  |  |  |  |
| 김병진 | |  |  |  |  |  |  |  |
| 이상기 | |  |  |  |  |  |  |  |
| 12/7 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 |  | | | | | | |
| 12/8 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |
| 12/9 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |
| 12/10 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |
| 12/11 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |
| 12/12 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |
| 12/13 | 강수빈 |  | | | | | | |
| 김병진 |  | | | | | | |
| 이상기 | 테스트 및 피드백 | | | | | | |