네트워크 게임 프로그래밍

프로젝트 기획서

[Escape Cave]

1. 2016182005 김경욱
2. 2018150002 권수민
3. 2018184009 박성준

목차

1. 게임 개요

가. 게임 정보

나. 조작 방법

다. 게임 플레이

2. High-Level Design

가. Lobby

나. In-Game

1) 서버

2) 클라이언트

3. Low-Level Design

가. 구조체

나. 함수

1) 서버

2) 클라이언트

4. 스레드

가. 구조

나. 동기화

5. 역할 분담

6. 개발 일정

가. 김경욱

나. 권수민

다. 박성준

1. 게임 개요
   1. 게임 정보

|  |  |
| --- | --- |
| 제목 | Escape Cave |
| 장르 | 협동 슈팅 게임 |
| 게임 인원 | 3명 |
| 스토리 | 3명의 플레이어가 힘을 합쳐 동굴을 탈출해 자유를 되찾기 위한 과정. 동굴 속 몬스터를 총탄으로 처치하여 게이트를 열고, 보스를 처치하여 동굴을 탈출하게 되는 이야기. |
| 개발 환경 | Visual studio 2019  OpenGL  TCP / IP  Windows Socket  Git, Github |

* 1. 조작 방법

|  |  |
| --- | --- |
| 상하좌우 키 | 전후좌우로 이동한다. |
| 'A' | 필드에 특수공격(폭탄)을 설치한다. |
| 'S' | 플레이어 공격 시의 총탄 수를 변경한다.  (1발 <-> 2발) |
| 마우스 클릭 | 총탄을 발사한다. |

* 1. 게임 플레이 규칙

- 3명의 플레이어가 함께 게임을 진행한다,

~~- 플레이어는 상하좌우 키로 이동하며 전방으로 총탄을 자동 발사한다.~~

- 플레이어는 상하좌우 키로 이동하며 마우스 클릭 시 총탄을 발사한다.

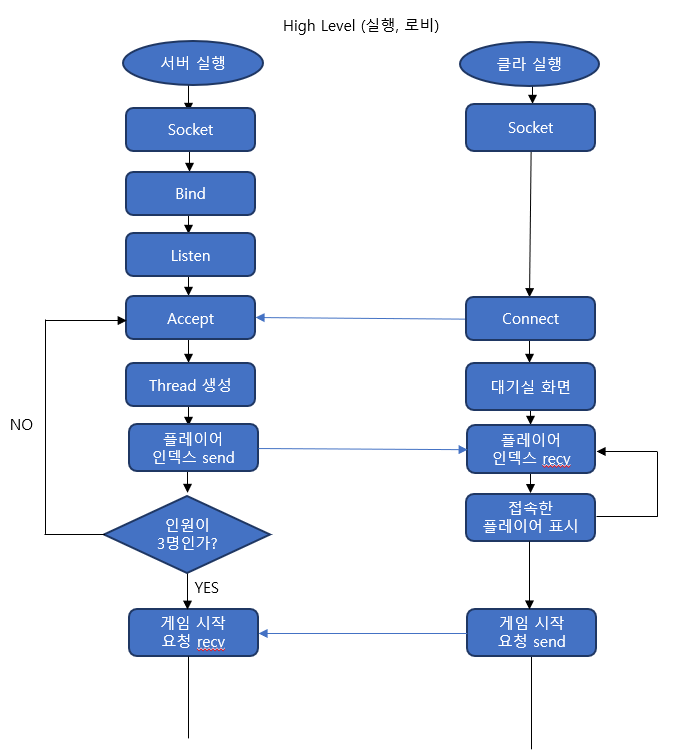
- 모든 총탄은 벽에 충돌 시 제거된다.

- 플레이어의 총탄은 몬스터와 충돌했을 경우, 몬스터의 총탄은 플레이어와 충돌했을

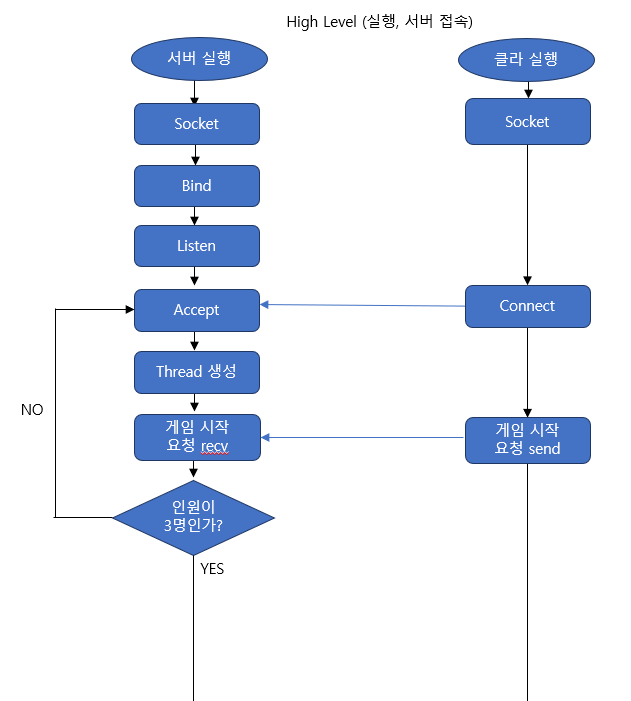
경우 제거된다.

- 한 명의 플레이어라도 생존한 상태에서 보스를 처치할 경우 게임에서 승리한다.

1. High-Level Design
   1. ~~Lobby~~
      1. ~~구조~~

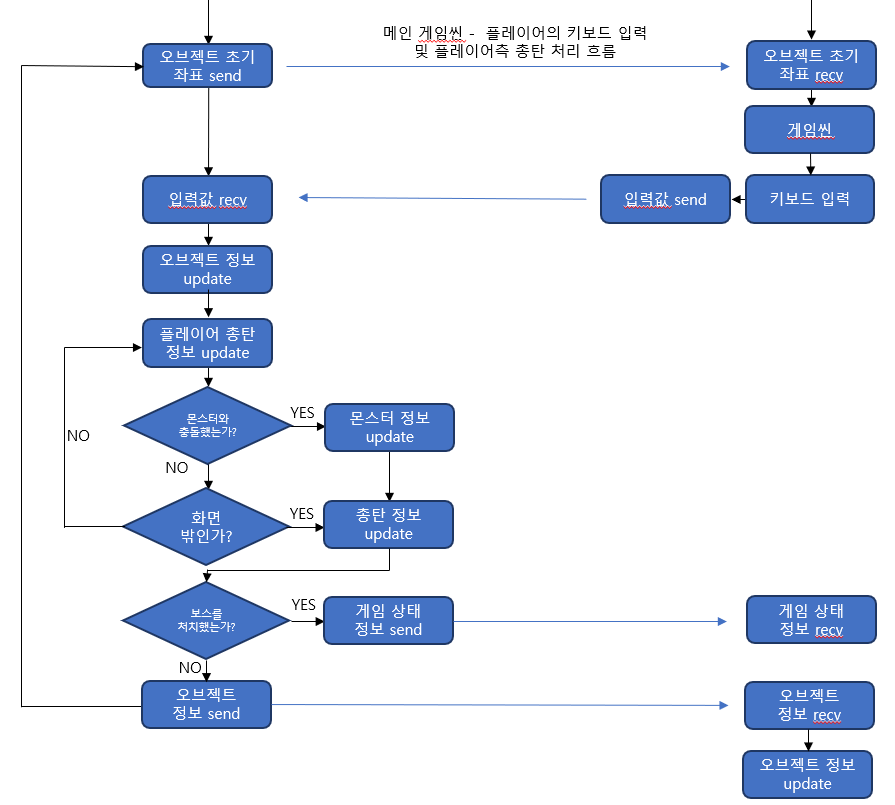


이하 구조로 변경, 로비 삭제됨.

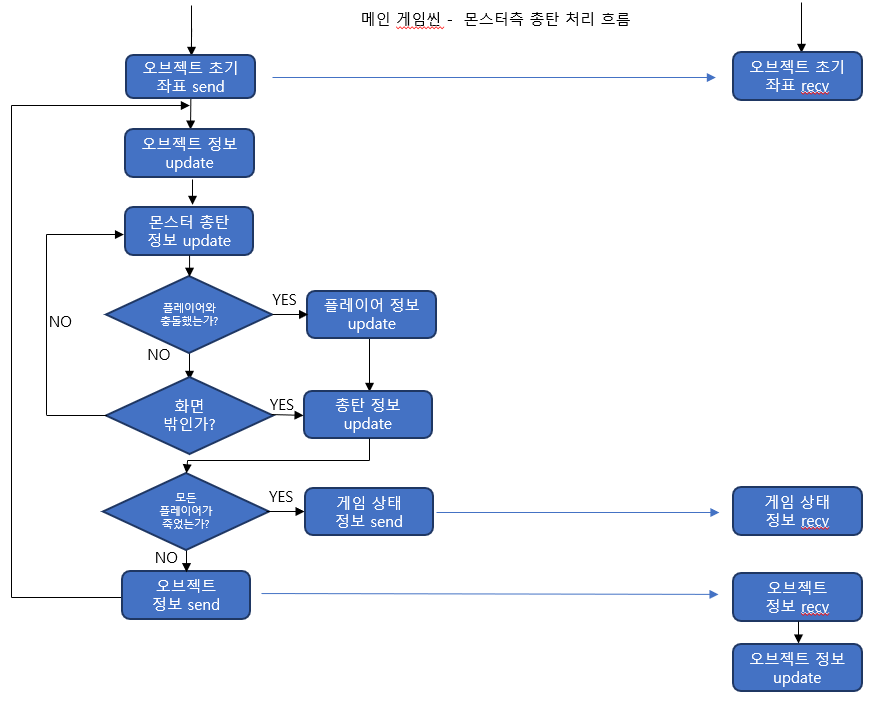


* + 1. 서버

1. 클라이언트의 연결 요청을 수신한다.
2. 3명의 클라이언트가 연결할 때까지 대기한다.
3. ~~클라이언트에게 모든 클라이언트의 정보를 송신한다.~~
4. ~~클라이언트로부터 게임 시작 요청을 수신한다.~~
   * 1. 클라이언트
5. 서버에게 연결 요청을 송신한다.
6. ~~서버로부터 접속한 모든 클라이언트의 정보를 수신한다.~~
7. ~~접속한 모든 클라이언트의 정보를 화면에 출력한다.~~
8. ~~서버에게 게임 시작 요청을 송신한다.~~
   1. In-Game
      1. 구조



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



* + 1. 서버

1. 클라이언트로부터 키 입력을 수신하여 게임 내 변화를 처리한다.
2. 클라이언트에게 초기 값의 오브젝트 정보 구조체를 송신한다.
3. 클라이언트에게 동기화된 오브젝트 정보 구조체를 송신한다.
4. 클라이언트에게 보스를 처치한 경우 승리 정보를, 모든 플레이어가 사망할 경우 패배 정보를 송신한다.
5. 오브젝트 정보를 게임 내 변화에 대응하여 갱신한다.
   * 1. 클라이언트
6. 플레이어가 입력한 키 입력 값을 서버에 송신한다.
7. 서버로부터 오브젝트 정보 구조체를 수신하여 객체를 생성, 초기화한다.
8. 서버로부터 동기화된 오브젝트 정보 구조체를 수신하여 좌표 및 상태를 갱신한다.
9. 서버로부터 승리/패배 정보를 수신하여 화면에 출력한다.
10. Low-Level Design
    1. 구조체
11. 가변 길이 데이터 전송을 위해 고정 길이로 보내는 구조체

typedef struct DataInfo {

char infoindex; // 패킷 타입

char datasize; // 패킷 크기

};

- 오브젝트 좌표 리스트: 모든 오브젝트의 좌표를 종류별로 분할하여 하나의 리스트에 저장한다.

typedef list<glm::vec3> SEND\_OBJECT\_LIST

SEND\_OBJECT\_LIST s\_ObjectList[OBJID::END];

1. ~~오브젝트 정보 구조체(플레이어, 몬스터, 총탄, 게이트)~~

~~typedef struct ObjectInfo {~~

~~vector<glm::vec3> Translate; //오브젝트의 좌표 이동 위한 벡터값 저장~~

~~char hp;~~  ~~//오브젝트의 남은 체력. 0일 시 오브젝트 제거~~

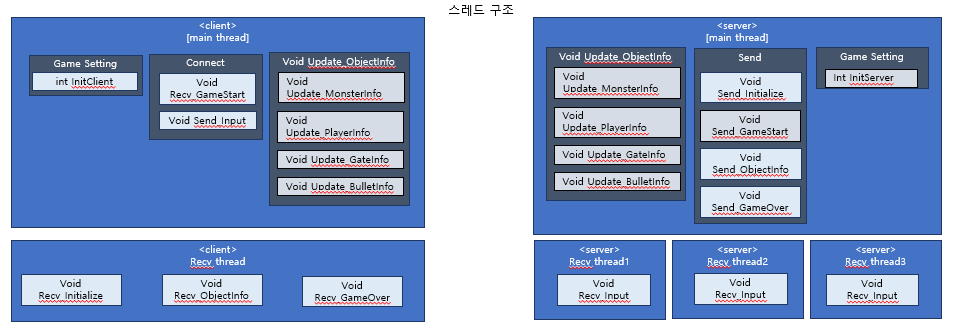
~~}~~

* 1. 함수
     1. 서버

1. Int InitServer() : 소켓을 생성하여 클라이언트의 접속 요청을 처리하는 함수
2. DWORD WINAPI RecvThread(): 클라이언트 메시지 수신용 스레드.
3. Void Send\_ObjectInfo(): 클라이언트에게 모든 오브젝트의 정보를 송신하는 함수
4. Void Send\_GameOver(): 클라이언트에게 게임 결과 및 오브젝트 정리 명령을 송신하는 함수
5. Void Send\_Initialize(): 클라이언트에게 게임 시작 시의 모든 오브젝트의 초기화 요청을 송신하는 함수
6. Void Recv\_Input(): 클라이언트의 키보드 입력을 수신하는 함수
7. Void Update\_ObjectInfo(): 모든 오브젝트의 정보를 게임 변화에 따라 갱신하는 함수
8. Void Update\_PlayerInfo(): 모든 플레이어의 정보를 게임 변화에 따라 갱신하는 함수
9. Void Update\_MosnterInfo(): 모든 몬스터의 정보를 게임 변화에 따라 갱신하는 함수
10. Void Update\_GateInfo(): 모든 게이트의 위치를 게임 변화에 따라 갱신하는 함수
11. Void Update\_BulletInfo(): 모든 총탄의 위치를 게임 변화에 따라 갱신하는 함수
12. Void Send\_GameStart() : 클라이언트에게 게임 시작 요청을 송신하는 함수
13. ~~Void Send\_ClientInfo() : 클라이언트에게 서버와 접속한 클라이언트의 정보를 송신하는 함수~~
14. ~~Void Recv\_GameStart() : 클라이언트로부터 게임 시작 요청을 수신하여 처리하는 함수~~
15. ~~Void RunServer() : 클라이언트의 접속부터 게임 시작까지를 담당하는 함수. 로비에 클라이언트가 2명 접속하고 방장이 게임 시작을 요청하면 게임을 시작.~~
    * 1. 클라이언트
16. int InitClient() : 소켓을 생성하여 서버와 연결하는 함수
17. DWORD WINAPI RecvThread(): 서버 메시지 수신용 스레드.
18. Void Send\_Input(): 키보드 입력을 서버로 송신하는 함수
19. Void Recv\_ObjectInfo(): 서버로부터 모든 오브젝트의 정보를 수신하는 함수
20. Void Recv\_GameOver(): 서버로부터 게임 결과 및 오브젝트 정리 명령을 수신하는 함수
21. Void Recv\_Initialize(): 서버로부터 요청을 받아 게임 시작 시의 모든 오브젝트의 초기 좌표, 상태를 초기화하는 함수
22. Void Update\_ObjectInfo(): 모든 오브젝트의 정보를 갱신하는 함수
23. Void Update\_PlayerInfo(): 모든 플레이어의 정보를 갱신하는 함수
24. Void Update\_MosnterInfo(): 모든 몬스터의 정보를 갱신하는 함수
25. Void Update\_GateInfo(): 모든 게이트의 정보를 갱신하는 함수
26. Void Update\_BulletInfo(): 모든 총탄의 정보를 갱신하는 함수
27. Void Recv\_GameStart() : 서버로부터 게임 시작을 요청을 받는 함수
28. ~~Void Recv\_ClientInfo() : 서버로부터 서버와 접속한 클라이언트의 정보를 수신하는 함수~~
29. ~~Void Update\_ClientInfo() : 모든 클라이언트의 정보를 화면에 출력하는 함수~~
30. ~~Void Send\_GameStart() : 서버에게 게임 시작을 요청하는 함수~~
31. 스레드
    1. ~~구조~~

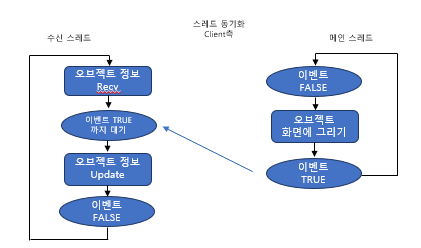
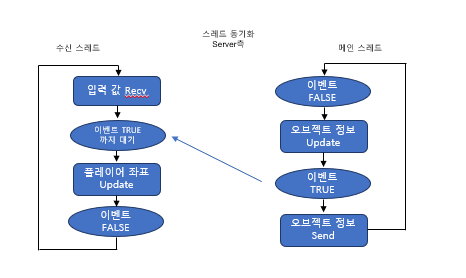
'

이하 구조로 변경됨

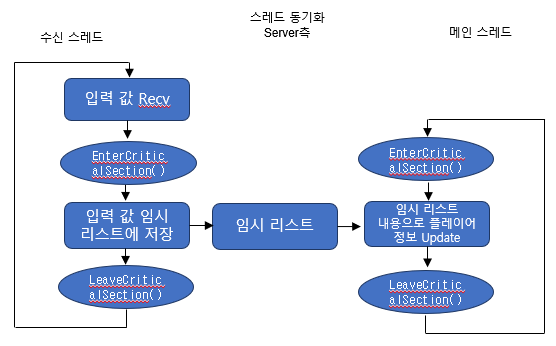


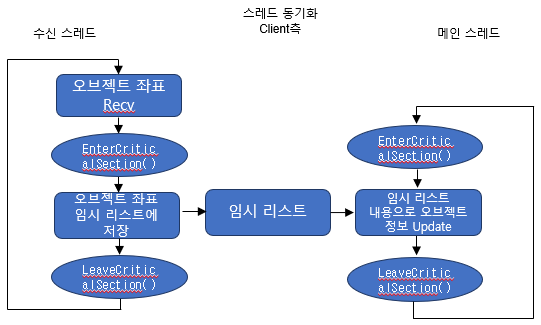
* 1. 동기화

클라이언트의 키 입력이 오브젝트 정보 업데이트 중 잘못 반영되는 상황을 막기 위해 오브젝트의 정보를 업데이트하는 동안에는 클라이언트로부터의 키 입력을 반영하지 않는다.



이벤트 대신 임계영역 사용





1. 역할 분담

|  |  |
| --- | --- |
| 김경욱 | 몬스터 정보 송수신 함수 등 몬스터 관련 함수 구현 |
| 권수민 | 플레이어 정보 송수신 함수 등 플레이어 관련 함수 구현 |
| 박성준 | 총탄 정보 송수신 함수 등 총탄 관련 함수 구현 |

1. 개발 일정
   1. 박성준

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 월 | 화 | 수 | 목 | 금 | 토 | 일 | |
| ~11/1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| 기획서  작성 및  보완 | 토의및점검 |  | client-server 연결  InitServer() InitClient() | 게임 시작 요청 처리  Send\_GameStart()  Recv\_GameStart()  RunServer() | 오브젝트 초기화 요청 송수신  Send\_Initialize() Recv\_Initialize() | |  |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 | |
|  | 토의및점검 |  | 객체/구조체 정리 및 보조 |  | 스레드 구조, 구조체 수정 및 보조 |  | |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 | |
|  | 토의및점검 |  | 총알 패킷 send, recv  총알 좌표 계산 |  | 총알, 몬스터충돌 체크 |  | |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 | |
|  | 토의및점검 |  | 총알, 플레이어충돌 체크 |  | 총알,벽 충돌체크 | Project Progress Report 작성 | |
| 29 | 30 | 12/1 | 2 | 3 | 4 | 5 | |
| 테스트 |  |  | 버그수정 | 버그수정 |  | 최종 토의 | |
| 6 | 7 |  | | | | | |
| 레포트 작성 | 발표 |

* 1. 김경욱

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 월 | 화 | 수 | 목 | 금 | 토 | 일 |
| ~11/1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 기획서 작성 및 보안 | 토의 및 점검 |  | client-server 연결  InitServer() InitClient() | 게임 시작 요청 처리  Send\_GameStart()  Recv\_GameStart()  RunServer() | 오브젝트 초기화 요청 송수신  Send\_Initialize() Recv\_Initialize() |  |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|  | 토의 및 점검 |  | 서버에서 몬스터 생성 및 데이터 송신  클라이언트에서 업데이트 | | |  |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  | 토의 및 점검 |  | 몬스터의 이동 경로 생성 및 데이터 송신  클라이언트에서 업데이트 | | | 몬스터 충돌 처리 확인 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
|  | 토의 및 점검 |  | 몬스터 배치 및 난이도 조절 | | | Project Progress Report 작성 |
| 29 | 30 | 12/1 | 2 | 3 | 4 | 5 |
|  | 토의 및 점검 | 버그 수정 | | | | 마지막 점검 |
| 6 | 7 |  | | | | |
|  | 발표 |

* 1. 권수민

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 월 | 화 | 수 | 목 | 금 | 토 | 일 |
| ~11/1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 기획서  작성 및  보완 | 토의 |  | client-server 연결, 스레드 생성  InitServer()  InitClient()  Recv\_Thread() | | 게임 시작 요청 처리  Send\_GameStart()  Recv\_GameStart() | |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|  | 토의 |  | 오브젝트 초기화 요청 송수신  RunServer()  Send\_Initialize()  Recv\_Initialize() | | client 키보드 입력 값 송수신Send\_Input() Recv\_Input() | 서버에서 Player 정보 갱신Update\_PlayerInfo() |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
|  | 토의 |  | 스레드 동기화 구현 | 갱신한 Object정보 client에게 송신  Send\_Objectnfo() Recv\_ObjectInfo() | 수신한 정보로 Player 정보 갱신Update\_PlayerInfo() | 브랜치 병합:  플레이어 통신 + 총탄 통신 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
|  | 토의 | 브랜치 병합: 서버-클라이언트 오브젝트 동기화 | 게임 결과 및 오브젝트 정리 요청  Send\_GameOver() Recv\_GameOver() | |  | |
| 29 | 30 | 12/1 | 2 | 3 | 4 | 5 |
|  | 토의 | 브랜치 병합: 플레이어 3인으로 확장 |  | 버그 수정 | | 병합 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|  | Project Progress Report 작성 |  |  | 버그 수정  임계영역 구현 | | 최종 토의 및 병합 |
| 13 | 14 |  | | | | |
|  | 발표 |