

## 네트워크 게임 프로그래밍 팀 프로젝트

게임공학과	2015156002	김근우
엔터테인먼트컴퓨팅	2016184037	하태웅
게임공학과	2017180006	김지영

김재경 교수님

## 목차

네트워크 게임 프로그래밍 팀 프로젝트 .....	1
1. 어플리케이션 기획 .....	3
1-1. 컨셉 .....	3
1-2. 게임 설명 .....	4
2. High Level Design .....	8
2-1. 기본 구조 .....	8
2-2. 기본 패킷 디자인 .....	9
3. Low Level Design .....	12
3-1. 클라이언트 .....	12
3-2. 서버 .....	15
4. 역할분담 .....	18
5. 개발환경 .....	18
6. 개발일정 .....	19

## 1. 어플리케이션 기획

### 1-1. 컨셉



**게임 제목** 2D FPS 게임

**게임 컨셉** 최후의 1인이 될 때 까지 경쟁하는 2D FPS 서바이벌 게임

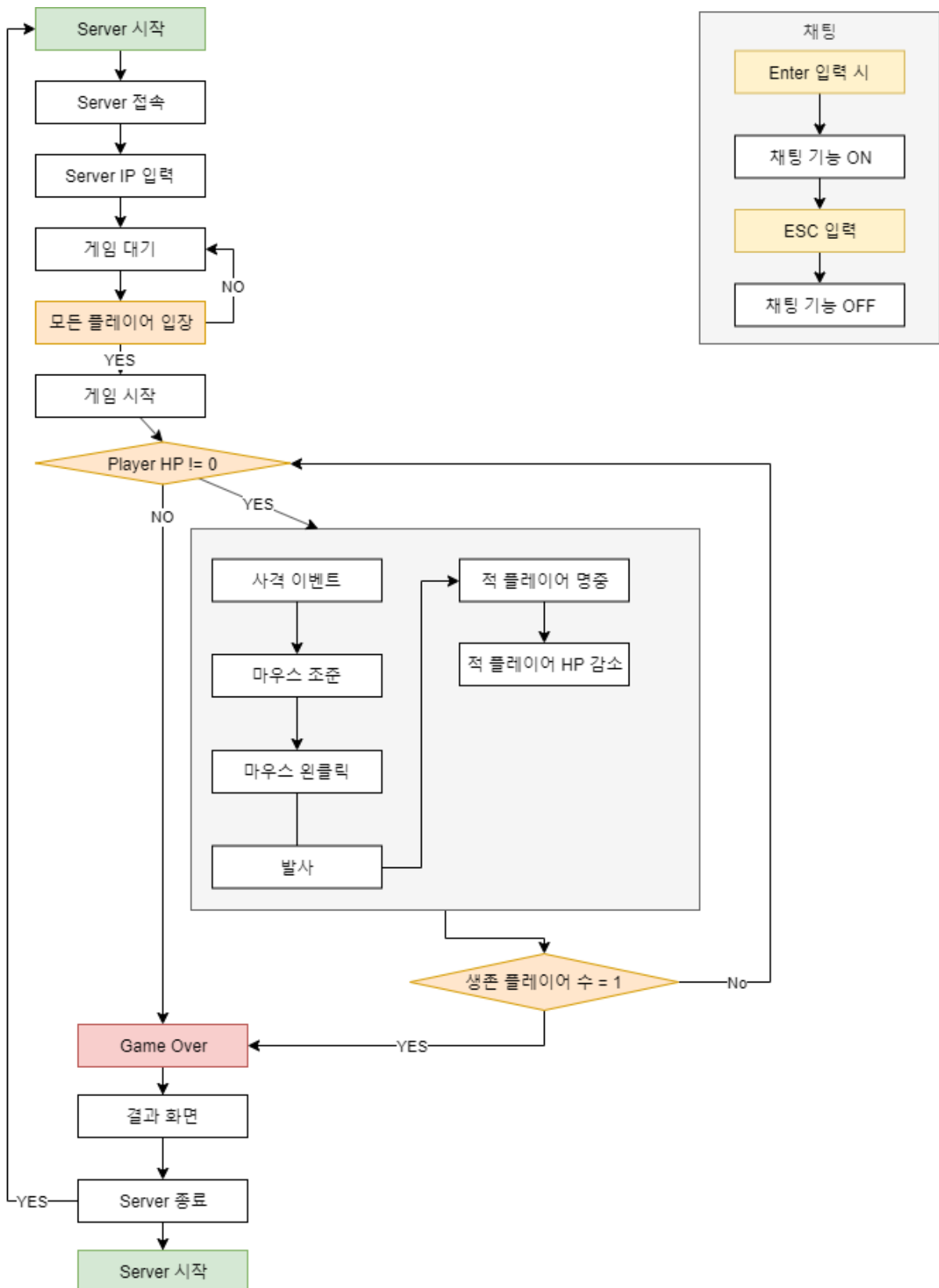
### 게임 소개

일정 지역 내에서 플레이어들은 서로 총을 발사하며 싸우게 된다.

총은 여러 종류가 존재하며, 플레이어는 아이템을 획득하여 총의 종류를 바꿀 수 있다.

## 1-2. 게임 설명

### A 게임 흐름



## B 게임 실행 방법

플레이어 중 누군가가 서버 프로그램을 실행한다.

이후 나머지 플레이어들이 클라이언트를 실행하고, IP를 입력하여 개설된 서버에 접속한다.

모든 플레이어가 접속하면 자동으로 게임이 시작된다.





## C 게임 규칙

기본 규칙	모든 플레이어가 입장 시 게임 시작
	플레이어는 HP 0가 되면 GAME OVER
	생존 플레이어 수가 1이 될 때까지 게임이 진행됨
사격 이벤트	마우스로 원하는 위치를 조준할 수 있음
	마우스 왼 클릭을 할 시 조준 한 위치에 탄환이 발사됨
무기 교체	필드에 무기가 존재함
	아이템이랑 플레이어가 충돌하면 해당 무기로 변경되어서 장착
플레이어 킬	플레이어는 공격(조준 후 탄환 발사)액션으로
	상대 플레이어의 HP를 깎을 수 있음
	적 플레이어 HP를 0로 만들면 Player Kill Count +1 / Alive Count -1 로 게임 내 수치가 변경됨

#### D 인 게임 표시 내용 (UI)

이름	설명
생존 표시	플레이어가 얼마나 남았는지 표시 Ex. 10명 생존 시 → 19 Alive
킬 표시	1 타 플레이어의 죽음을 표시하는 UI 2 플레이어 킬에 사용된 총도 함께 <b>전체 공지</b> 함 Ex. LOL이 AK-47을 사용해서 Kevin을 죽임 → LOL killed Kevin with AK-47
HP	플레이어의 남은 <b>HP 양</b> 을 표시함
사용하는 총	플레이어가 현재 사용하는 총은 플레이어 얼굴에 표시

#### E 게임 오브젝트

이미지	이름	설명
	플레이어	Player 오브젝트 HP: 100 무기를 드롭해 상대 플레이어의 HP를 내릴 수 있음
	적 플레이어	적 플레이어 플레이어를 공격함
	무기	무기는 3 종류가 있음 권총/라이플/저격총
	엄폐물	플레이어의 공격을 막을 수 있는 엄폐물

## F 조작 방법

### - *마우스 Mouse*



\* **MOUSE MOVE** 마우스 커서를 움직여서 원하는 방향으로 총을 쏠 수 있음

액션	조작 키
총 조준 움직임	Move
발사	L click

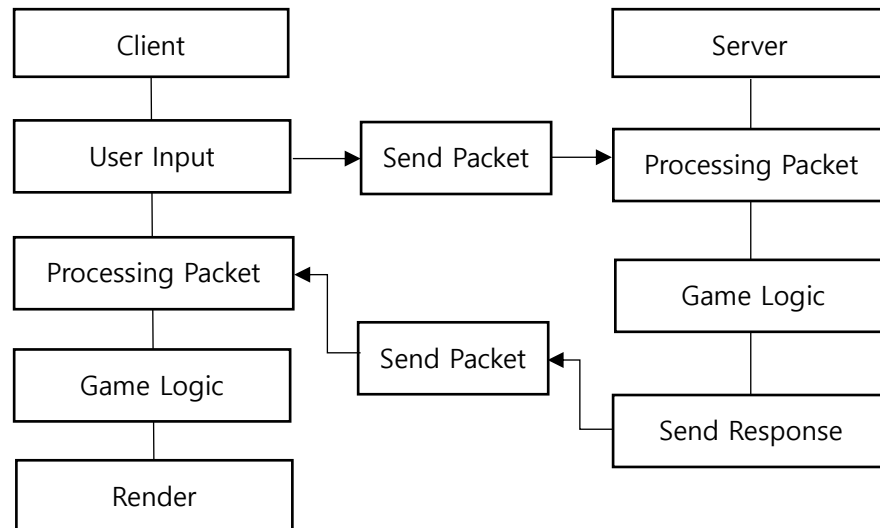
### - *키보드 Keyboard*

액션	조작 키
캐릭터 이동	W, A, S, D
채팅 시작	Enter
채팅 종료	ESC

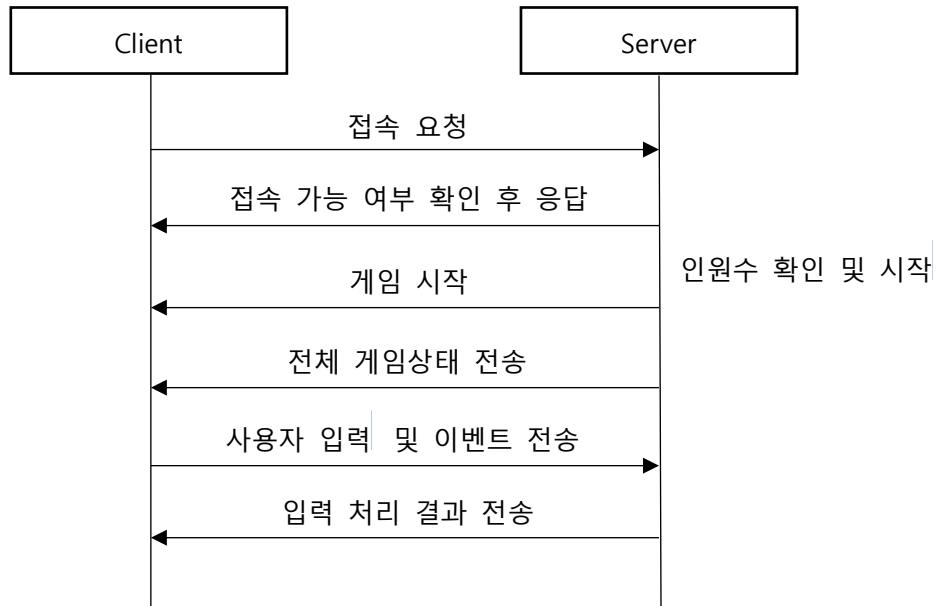
## 2. High Level Design

### 2-1. 기본 구조

#### - 서버/클라이언트 역할도



#### - 클라이언트와 서버 간 통신



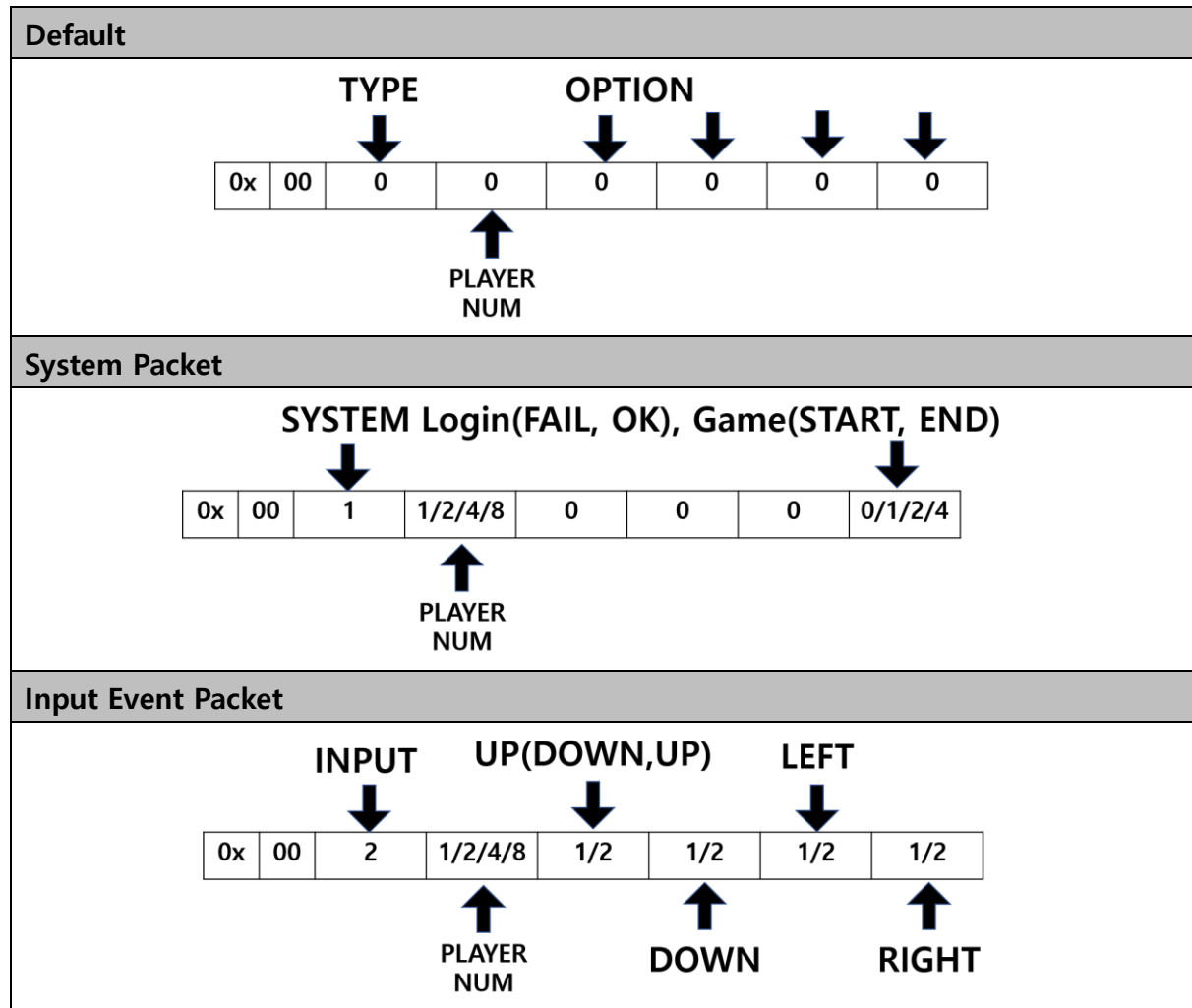
통신을 위한 프로토콜은 TCP를 사용한다. 캐릭터, 총알, 아이템 등 오브젝트들은 동기화를 위한 고유한 ID를 갖는다.

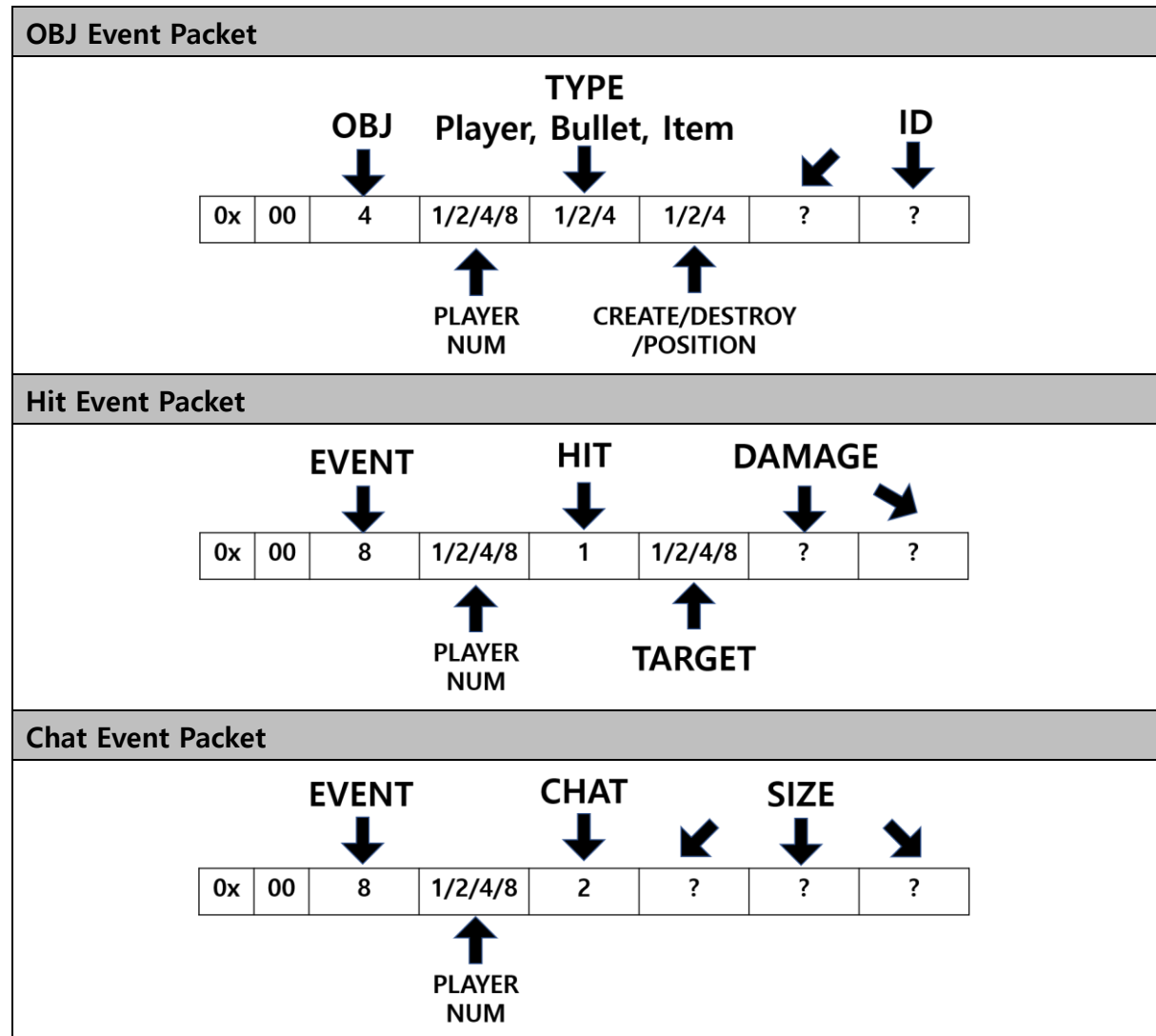


## 2-2. 기본 패킷 디자인

고정길이의 패킷을 처리하고 크기에 맞게 4Byte Int 정수형 변수에 bit 단위로 옵션을 넣어준다.

부가정보(위치, 방향)가 필요한 패킷인 경우 추가적인 Recv를 호출한다.





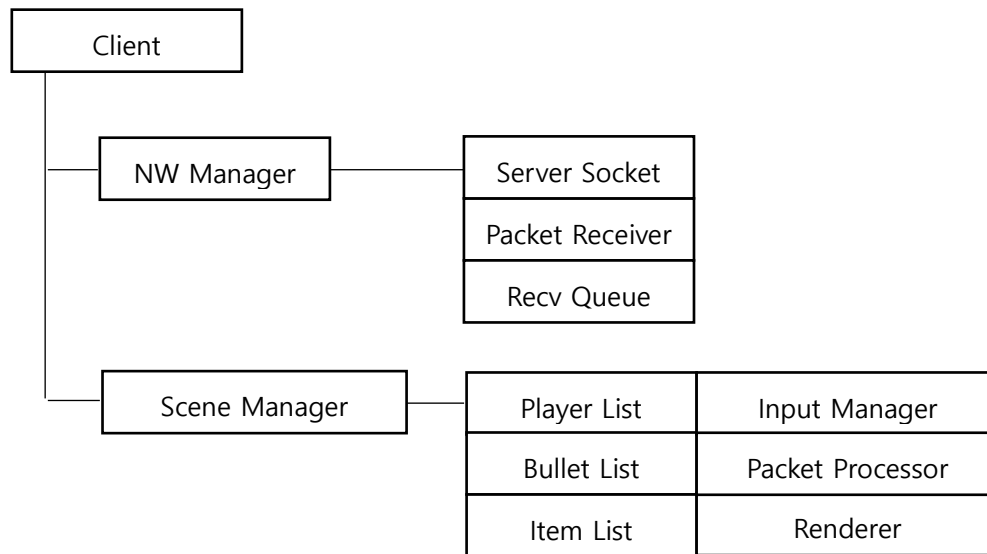
TYPE	OPTION	16진수
<b>Default</b>	PLAYERNUM	0x0?0000 (0x010000, 0x020000, 0x040000, 0x080000)
<b>SYSTEM</b>	SYSTEM	0x1?0000
	LOGIN FAIL, OK	0x1?0000, 0x1?0001
	START, END	0x100002, 0x1?0004 //END에 경우 PLAYERNUM이 WINNER
<b>INPUT</b>	INPUT	0x2?0000
	UP (Down, Up)	0x2?1000, 0x2?2000
	DOWN	0x2?0100, 0x2?0200,
	LEFT	0x2?0010, 0x2?0020,
	RIGHT	0x2?0001, 0x2?0002,
<b>OBJ</b>	OBJ	0x4?0000 //PLAYERNUM 행위 주체.
	TYPE (Player, Bullet, Item)	0x4?1000, 0x4?2000, 0x4?4000
	CREATE	0x4??100
	DESTROY	0x4??200
	POSITION 갱신	0x4??400
	ID(ITEM)	0x4??4??
<b>EVENT</b>	EVENT	0x8?0000
	HIT	0x8?1000
	TARGET	0x801?00
	DAMAGE	0x801???
	CHAT	0x8?2???

## 패킷 디자인 예시

사용자 1의 LEFT\_DOWN 키 입력 → 0x210020

사용자 3의 Bullet 생성 → 0x441200

### 3. Low Level Design



#### 3-1. 클라이언트

##### A 자료구조

List<int> RecvQueue	서버에서 받은 패킷을 보관
List<Vec2f> Vec2Queue	서버에서 받은 Vec2F 패킷 보관
map<int, obj> PlayerList	플레이어 관리 객체
map<int, obj> BulletList	총알 관리 객체
map<int, obj> ItemList	아이템 관리 객체

## B Scene Mgr

Void CollisionCheck();	Player 객체의 충돌 체크 이벤트
Void Update(float eTime)	Tick 당 Scene 업데이트 함수
int ProcessPacket(list<int> &RecvQueue);	수신 큐에 담겨 있는 패킷들을 처리하여 게임에 반영
void RenderScene()	화면에 렌더링
입력 이벤트	
void KeyUpInput(u_char key, int x, int y)	
void KeyDownInput(u_char key, int x, int y)	
void MouseUpInput(u_char key, int x, int y)	
void MouseDownInput (int button, int state, int x, int y)	
int MakeMessage( int type, int x, int y)	발생된 이벤트 들을 메시지 패킷으로 만들어 서 NW Manager에게 전달
오브젝트 관리	
Void CreateObj (char obj_type, char obj_id, Vec2f float y)	
void DeleObj( Obj )	

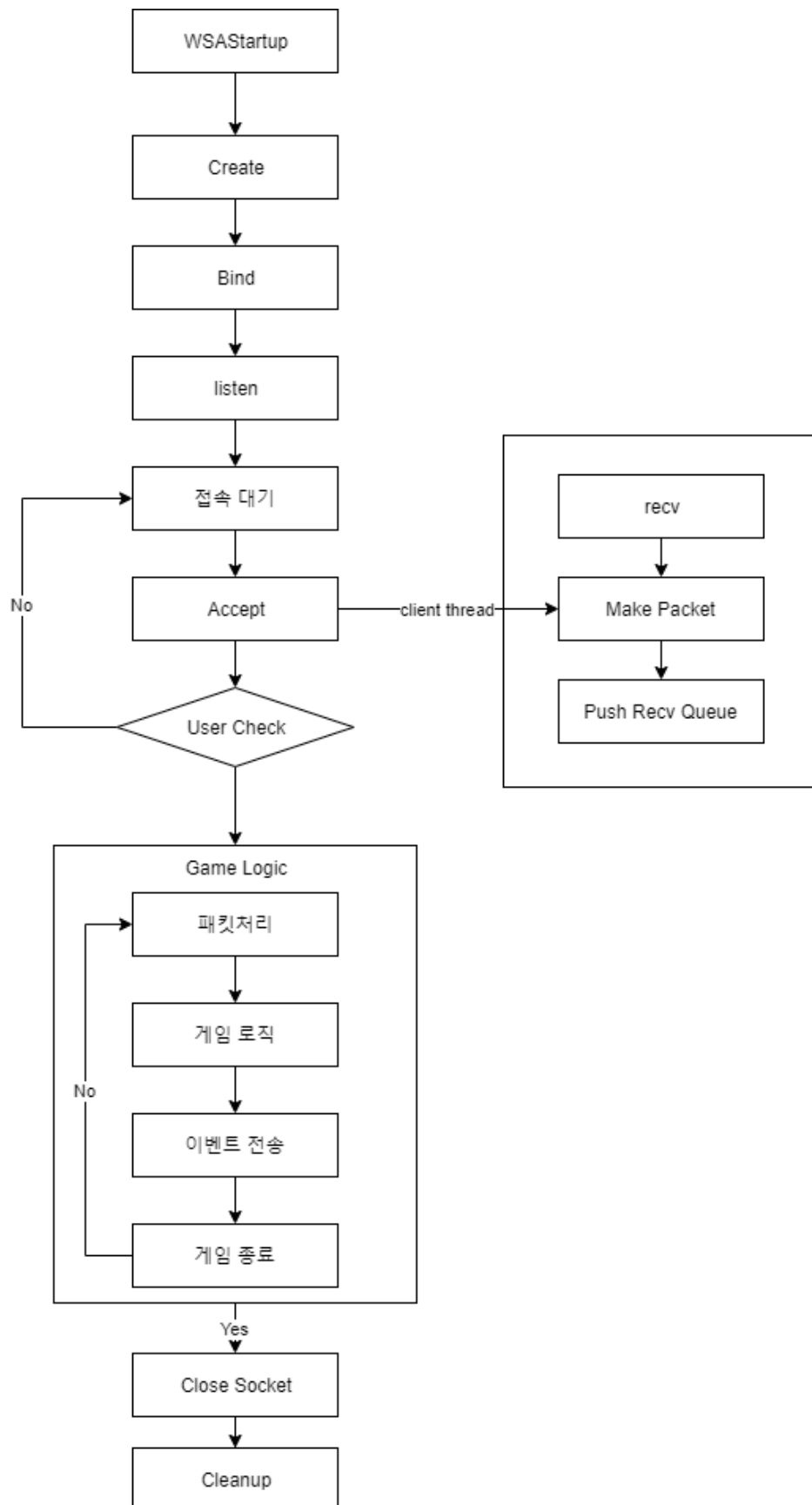
### C Object

Void Update(float eTime)	객체 업데이트 함수
Func Get/Set	
Vec2f mPos	위치
Vec2f mSize	크기
Vec2f mVel	속도
Vec2f mAcc	가속도
float mFricCoef	마찰 계수

### D NW Mgr

SOCKADDR_IN mServerAddr	서버 주소 구조체
HANDLE mThread	쓰레드 관리 핸들
void Init()	IPinput(), sock(), connect()
DWORD WINAPI MessageReceiverFunc (LPVOID ClientSocket)	지속적으로 SendQueue 에 담겨있는 패킷들을 Send 하고 서버에서 보내는 패킷들을 Recv 하는 쓰레드
Int RecvPacket( int )	패킷을 받아 Recv 큐에 삽입
int SendPacket( int )	Send 큐에 담겨져 있는 패킷들을 전송
bool On(int PACKET, PACKET_FLAG PF) {return (PACKET & PF) == PF;}	패킷 옵션 처리 함수

### 3-2. 서버



## A 자료구조

List<int> RecvQueue	클라이언트들이 송신한 패킷을 보관하는 리스트
List<int> SendQueue	클라이언트에 보낼 패킷을 보관
map<int, obj> PlayerList	플레이어 관리 객체
map<int, obj> BulletList	총알 관리 객체
map<int, obj> ItemList	아이템 관리 객체

## B 서버 함수

void CreateObject(char obj_type, char obj_id, float x, float y)	오브젝트를 특정 위치에 생성하는 함수
void DestroyObject(char obj_type, char obj_id)	특정 오브젝트를 제거하는 함수
DWORD WINAPI MessageReceiverFunc(LPVOID ClientSocket)	클라이언트 소켓을 입력받아 지속적으로 Recv 를 수행하는 스레드함수
void MakePacket()	내부적으로 recv 를 반복하며, 패킷을 완성하는 함수, 완성 시 PushRecvQueue 함수로 RecvQueue 에 패킷을 삽입한다.
bool StartCheck(int num)	모든 플레이어가 접속했는지 반환
void ProcessPacket(list<int>& RecvQueue)	RecvQueue 에 쌓인 패킷 처리 게임에 반영하는 함수
void Update(float fElapsedTime)	서버 내 게임월드의 오브젝트들을 갱신하고, 충돌 처리 등 다양한 처리를 수행하는 함수
int send_packet(int client, void* packet)	패킷을 클라이언트에게 전송하는 함수, 다른 패킷 함수들의 기본이 된다.
int send_login_ok_packet(int client)	클라이언트가 접속 가능함을 알리는 패킷을 전송하는 함수
int send_login_fail_packet(int client)	클라이언트가 접속 불가능함을 알리는 패킷을 전송하는 함수, 이후 접속 종료처리를 수행한다.
int send_game_start_packet(int client);	게임 시작을 클라이언트에게 알리는 함수

(뒷장에 이어서 표시)



int send_game_end_packet(int client);	게임 종료를 클라이언트에게 알리는 함수
int send_chat_packet(int client, char* message)	채팅 내용을 클라이언트에게 전송하는 함수
int send_player_pos_packet(int client, int who)	특정 플레이어의 위치를 클라이언트에게 전송하는 함수
int send_player_use_item_packet(int client, int who, char item)	특정 플레이어가 아이템을 획득한 것을 클라이언트에게 전송하는 함수
int send_hitted_character_packet(int client, int who, int damage)	특정 플레이어가 데미지를 입은 것을 클라이언트에게 전송하는 함수
int send_create_obj_packet(int client, char obj_type, char obj_id, float x, float y)	특정 위치에 오브젝트를 생성하는 패킷을 전송하는 함수
int send_destroy_obj_packet(int client, char obj_type, char obj_id)	특정 오브젝트를 제거하는 패킷을 전송하는 함수
bool On(int PACKET, PACKET_FLAG PF) {return(PACKET & PF) == PF}	패킷 옵션 처리 함수

#### 4. 역할분담

##### [네트워크]

구분	내용		담당
네트워크	N/W Mgr 구현		김근우
	수신 패킷 처리 구현		하태웅
	Event 송신 패킷 처리		김지영
	패킷 구분 함수		하태웅

##### [클라이언트]

구분	내용		담당
클라이언트	OBJ_PLAYER 구현		김근우
	OBJ_ITEM 구현		하태웅
	OBJ_BULLET 구현		김지영
	결과, 대기화면 구현		김지영
	Collision Check 구현		하태웅
	입력처리 구현		김근우
	리소스 수집		공통

#### 5. 개발환경

Platform	PC / Windows
IDE	Visual Studio 2019 Community
VCS	Git
Language	C++ / C
WinSock Version ( Protocol )	2.2 / TCP
Graphics Library	OpenGL

## 6. 개발일정

### - 1주차 (10/29 ~ 11/3)

	10/28	10/29	10/30	10/31	11/1	11/2	11/3
김근우	계획서 초안 회의		회의 검수	리소스 수집			
하태웅							
김지영							

### - 2주차 (11/4 ~ 11/10)

	11/4	11/5	11/6	11/7	11/8	11/9	11/10
김근우	진행상황 확인 및 회의	서버 기초프레임 워크 구현	진행상황 확인 및 회의	서버 기초 프레임 워크 구현			
하태웅		클라 프레임워크 수정		클라 프레임워크 수정			
김지영							

### - 3주차 (11/11 ~ 11/17)

	11/11	11/12	11/13	11/14	11/15	11/16	11/17
김근우	진행상황 확인	플레이어 객체 구현 시작	진행상황 확인	플레이어 객체 물리 구현	플레이어 객체 텍스 처 적용		
하태웅		패킷 구분 함수 구현		패킷구분 함수 구현			
김지영	연구실 출장						

2019-10-29 네트워크 게임 프로그래밍 팀 프로젝트

- 4주차 (11/18 ~ 11/24)

	11/18	11/19	11/20	11/21	11/22	11/23	11/24
김근우	진행상황 확인 및 회의	입력처리 구현	진행상황 확인 및 회의	입력처리 구현	단일 클라 Render 테스트		
하태웅		아이템 객 체 구현		충돌 체크 구현			
김지영		총알 객체 구현		Event 송 신 패킷 처리			

- 5주차 (11/25 ~ 12/1)

	11/25	11/26	11/27	11/28	11/29	11/30	11/31
김근우	진행상황 확인 및 회의	N/W Mgr 구현 및 정리	진행상황 확인 및 회의	N/W Mgr 구현 및 정리	서버-클라 전송 테스 트 (Render 미적용)		
하태웅		수신 패킷 처리 구현		수신 패킷 처리 구현			
김지영		Event 송 신 패킷 처리		결과/대기 화면 구현			

- 6주차 (12/2 ~ 12/8)

	11/25	11/26	11/27	11/28	11/29	11/30	11/31
김근우	진행상황 확인 및 회의	디버깅/ 오류 수정	서버/클라 접속테스 트	최적화/ 오류 수정/ 미흡 부분 수정			
하태웅							
김지영							

- 7주차 (12/9 ~ 12/11)

	12/9	12/10	12/11
김근우	최종 테스트	작업일지 정리 및 검수	최종제출
하태웅			
김지영			