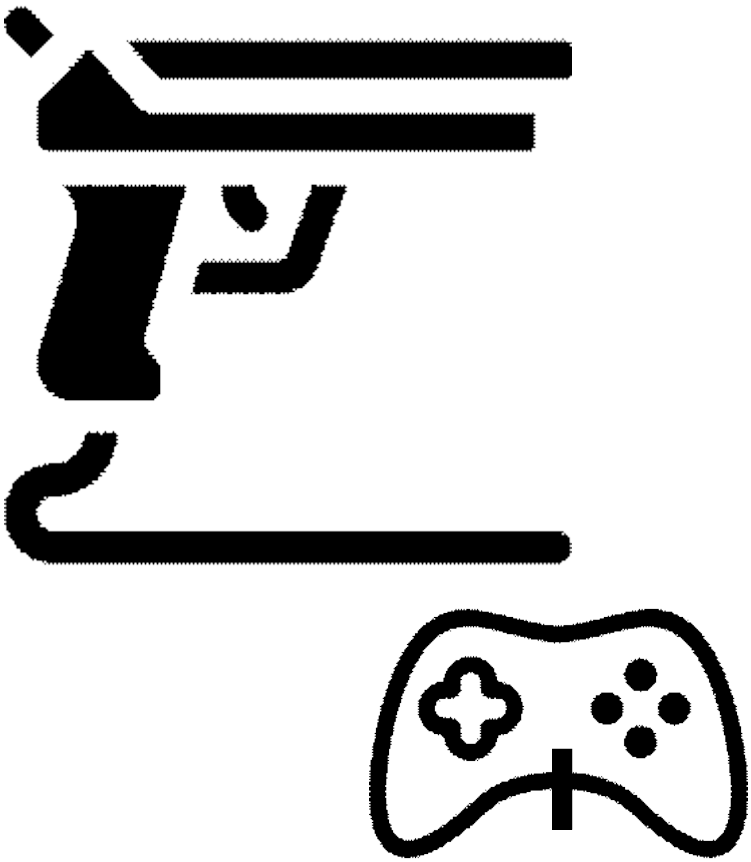
My little Shooting Game



[Revision history]

Revision date	Version #	Description	Author
	0.1	First Concept document	

= Contents =

1. Business purpose

1) Project background

오락이란 인간이 살아가는 데에 필수적인 인간의 기본 욕구로서 즐기고 싶다는 기본적인 감정을 일으키는 장치 산업이라 한다. 현대 사회에는 영화 만화책 드라마 예능프로 등의 여러 종류의 오락들이 있다. 하지만 이 중 어떠한 오락거리도 게임만큼 몰입하기 쉽고 신선한 충격을 안겨주진 못했다. 통계청의 정보에 따르면 취미 생활로 게임을 즐기는 사람들의 비중이 꽤 높다는 것을 알 수 있다. 하지만 막상 게임을 만들려고 해도 어떤 장르의 게임을 만들지에 대한 고민에 부딪히게 된다. 단순한 2D 퍼즐 계열의 게임부터 웅장한 서사의 주인공이 되어 세계를 탐험하는 rpg 계열의 게임도 있다.

몰입하기 쉬운 게임 중에서도 가장 몰입하기 쉬운 장르는 1인칭 시점을 가진 FPS이다. 초기의 FPS 게임은 그 당시 다른 장르들이 그랬듯이 직관적인 재미에 초점이 맞추어져 있었다. 하지만 몰입이 잘 되는 특징을 살려 점차 현실적이고 의미있는 스토리를 전달하는 FPS 게임도 많아졌다. 하지만 이에 반해 직관적인 재미는 점차 떨어지게 되었다. 이러한 점이 최근 FPS 게임에서 개인적으로 느낀 아쉬움이었다.

이러한 아쉬움에서 과거의 직관적인 재미를 추구하던 과거의 FPS 게임을 현대적으로 재해석해보자는 생각을 하게 되었다. 다른 요소들보다 “쏘고 맞춘다”라는 요소에 집중한 시스템을 만든다면, 과거와 같이 직관적인 재미를 느낄 수 있을 것이다.

그래픽 부분은 가벼운 느낌을 주는 복셀 그래픽을 활용하려고 한다. 고화질의 게임 그래픽이 주는 몰입감, 게임의 무게감이 있지만, 목표로 하는 게임의 테마와는 어울리지 않는다. 복셀 그래픽은 고전 게임의 픽셀 그래픽을 연상할 수 있는 등 목표로 하는 게임과 잘 어울리는 그래픽이 될 것이다.

또한 멀티플레이의 기능을 같이 구현하여 주변의 친구와 함께 즐길 수 있도록 한다. 멀티플레이 기능은 현대적으로 재해석된 요소지만, 함께 하는 게임은 과거의 오락실에서도 있던 개념이다. 멀티플레이 기능을 적절히 사용한다면 게임이 목표로 하는 과거의 향수에도 적합한 게임이 될 것이다.

그렇기에 주변 친구들과 가볍게 즐길 수 있는, 다시 말해 멀티플레이가 가능한 일인칭의 슈팅 게임을 만들고자 한다.

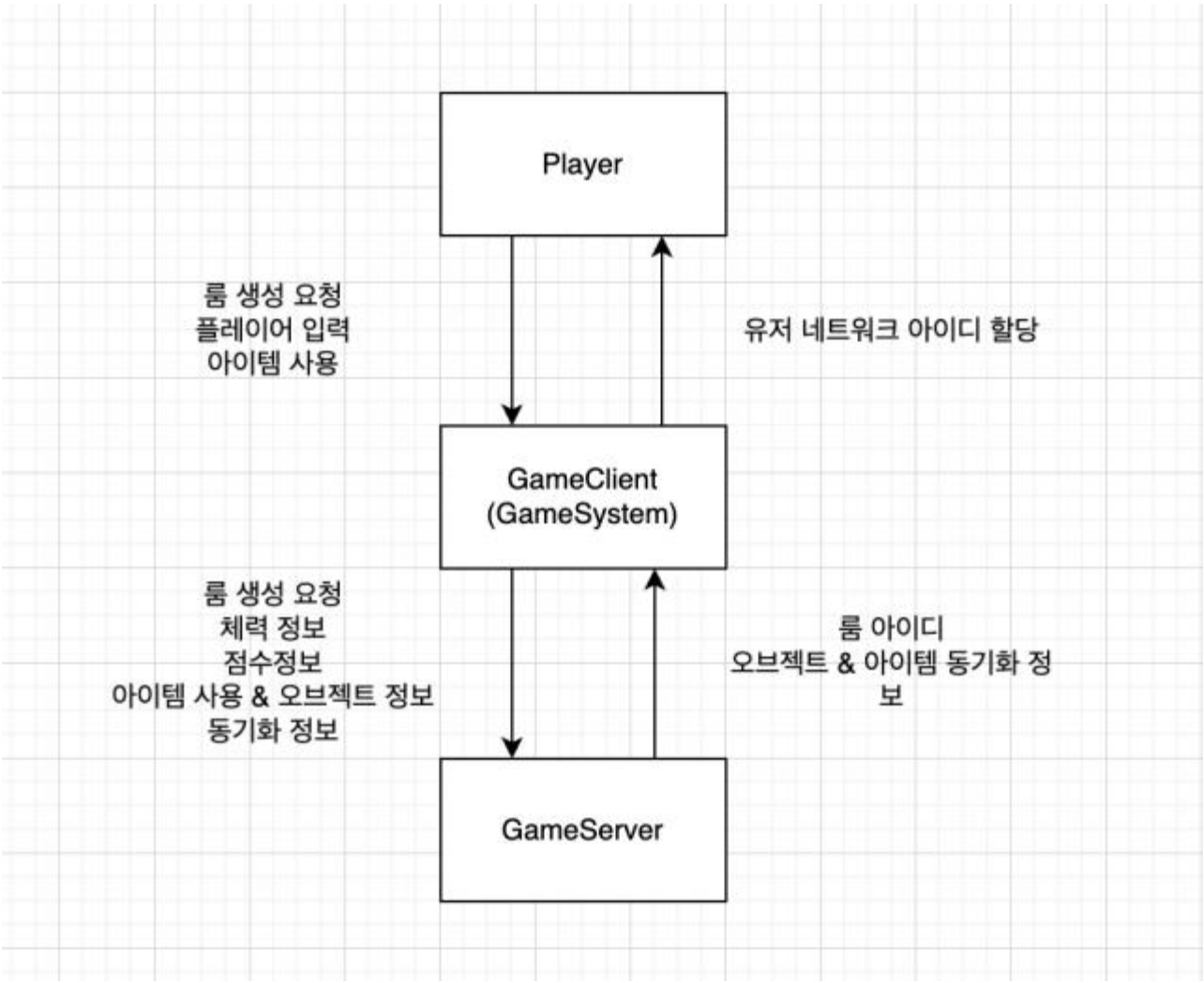
2) Goal

다른 플레이어와 정해진 룸에 접속하여 멀티플레이가 가능한 환경을 구성한다.
정해진 규칙을 따르며 간단한 게임이 가능하게 한다.

3) Target Market

주변 친구와 가볍게 게임을 즐기고 싶은 사람
평소 FPS 계열의 게임을 즐기는 게이머

2. System context diagram



3. Use case list

1) 랜덤 룸 생성

Actor	player(host, client)
Description	룸이 없는 상태에서 최초로 방에 들어온 사람이 host가 되고 룸의 최대 인원이 되기 전까지 들어온 사람은 client로 룸에 들어온다.

2) 룸 떠나기

Actor	player(host, client)
Description	게임 씬에서 다시 로비로 나온다. 만약 나온 플레이어가 호스트였을 경우 호스트 다음으로 들어온 플레이어에게 호스트를 위임한다.

3) 플레이어 입력

Actor	player, System(Gameclient)
Description	사용자로부터의 입력을 하나의 형식으로 변경한다. (키보드 마우스, 터치스크린, 조이패드 등의 입력을 정해진 형식으로 변형)

4) 플레이어 이동

Actor	System(Gameclient), Server
Description	사용자의 입력으로 이동한 케릭터를 서버에 동기화하여 다른 플레이어의 시점에서도 이동한다.

5) 체력 동기화

Actor	System(Gameclient), Server
Description	몹 오브젝트 혹은 상대 플레이어로부터 받은 피해를 서버에 동기화하여 다를 플레이어의 시점에서도 피해받은 정보를 공유한다.

6) 아이템 동기화

Actor	System(Gameclient), Server
Description	플레이어가 소유하고 있는 아이템(장비하고 있는 아이템)을 동기화하여 다른 플레이어의 시점에서도 같은 무기를 장비한 모습을 확인할 수 있게 한다.

7) 점수 동기화

Actor	System(Gameclient), Server
Description	현재 플레이어별 획득한 점수를 서버에 공유하여 현재 스코어를 확인할 수 있게한다.

8) 아이템 획득

Actor	System(Gameclient), player, server
Description	플레이어가 드롭된 아이템 위치로 이동하면 아이템을 획득하고 획득정보를 서버에 전송하여 다른 플레이어의 시점에서도 드롭된 아이템을 사라지게 한다.

9) 아이템 사용

Actor	System(Gameclient), player
Description	플레이어가 획득한 아이템을 사용하여 플레이어의 상태를 변화시킨다.

10) 오브젝트 처치

Actor	System(Gameclient), server
Description	오브젝트가 처치되면 현재 해당 오브젝트가 처치되었음을 서버에 알리고 오브젝트가 드롭하는 아이템을 추가한다. 추가한 아이템 정보를 서버에 전송하여 모든 플레이어가 드롭된 아이템에 대한 정보를 얻게 한다.

11) 무기 사용

Actor	System(Gameclient), player
Description	플레이어가 현재 장비 중인 무기를 사용하여 주변 오브젝트에게 피해를 준다.

4. Concept of operation

1) 서버에 접속

Purpose	멀티플레이를 원하는 모든 플레이어는 서버에 접속해야 함
Approach	서버에 접속하여 멀티플레이가 가능한 환경을 제공한다.
Dynamics	앱을 실행하여 서버에 접속할 경우
Goals	서버에 접속한다.

2) 랜덤 룸 생성

Purpose	다른 유저들과 멀티플레이를 하기 위해서는 서버에 접속하여 랜덤 룸에 입장하여야 한다.
Approach	사용자가 서버에 접속 후 랜덤 룸에 참여를 희망하면 해당 게임의 버전을 확인하고 룸에 참여 혹은 새로운 룸을 생성한다.
Dynamics	앱 실행 시 서버에 접속한 경우
Goals	랜덤 매칭 시스템을 구현한다.

3) 룸 떠나기

Purpose	현재 진행 중인 게임을 종료하고 싶거나 다른 유저들과 플레이 하고 싶다면 현재 룸을 떠나야 한다.
Approach	룸을 떠날 수 있게 한다.
Dynamics	서버에 접속되어 있고 현재 룸을 떠나고 싶은 경우
Goals	룸에서 떠나는 기능을 구현한다.

4) 플레이어 입력

Purpose	자신의 플레이어를 제어하기 위해서는 사용자로부터 입력을 받아야 한다.
Approach	사용자의 입력을 바탕으로 플레이어를 제어할 수 있게 한다.
Dynamics	게임에 접속하여 플레이어를 제어할 경우
Goals	사용자에 입력에 반응하는 플레이어 오브젝트를 만든다.

5) 체력 동기화

Purpose	모든 오브젝트는 같은 체력을 가지고 있어야 한다.
Approach	오브젝트의 체력을 서버에 전송한다.
Dynamics	오브젝트의 체력이 변화한 경우
Goals	오브젝트의 체력을 동기화 한다.

6) 아이템 동기화

Purpose	이미 사용된, 획득된 아이템은 게임 레벨에서 사라져야 한다. (하나의 아이템에서 하나의 아이템을 획득해야 한다.)
Approach	현재 아이템의 정보를 서버와 공유한다.
Dynamics	다른 플레이어에 의해 아이템이 사용 혹은 획득된 경우
Goals	현재 아이템의 정보를 모든 플레이어에게 공유한다.

7) 점수 동기화

Purpose	플레이어가 획득한 점수를 모든 플레이어에게 공유하여 현재 점수 현황을 알려야 한다.
Approach	플레이어별 점수를 서버에 공유한다.
Dynamics	점수에 변화가 발생한 경우
Goals	플레이어별 점수 현황을 공유한다.

8) 아이템 획득

Purpose	획득한 아이템 정보를 플레이어끼리 공유해야 한다. (새로운 무기를 획득한 경우 다른 플레이어도 해당 정보를 공유해야 한다.)
Approach	플레이어가 획득한 아이템의 정보를 서버에 공유한다.
Dynamics	다른 플레이어에 의해 아이템이 획득된 경우
Goals	아이템을 획득한 플레이어의 정보를 모든 플레이어에게 적용한다.

9) 아이템 사용

Purpose	소비성 아이템을 사용했을 때 변화한 플레이어 정보를 다른 플레이어에게도 적용해야 한다. (수류탄, 탄알 등의 아이템을 사용했을 때 다른 플레이어에게 해당 아이템이 소모되었다는 것을 알려야 함)
Approach	다른 플레이어에게 아이템이 사용되었다는 것을 알려준다.
Dynamics	다른 플레이어에 의해 아이템이 사용된 경우
Goals	사용된 아이템의 정보를 서버에 공유한다.

10) 오브젝트 처치

Purpose	오브젝트가 처치된다면 드롭된 아이템을 다른 플레이어에게도 알림
Approach	다른 플레이어에게 오브젝트가 처치되었다는 것을 알린다.
Dynamics	오브젝트가 처치된 경우
Goals	처치된 오브젝트 정보를 서버에 공유하고, 드롭된 아이템을 모든 플레이어에게 알린다.

11) 무기 사용

Purpose	플레이어가 무기를 사용하여 투사체 오브젝트가 생성됨을 알림
Approach	다른 플레이어에게 투사체 오브젝트를 보여줌
Dynamics	플레이어가 무기를 발사함
Goals	플레이어가 발사한 무기의 투사체를 다른 이에게 보여줌

5. Problem statement

이 장에서는 만들고자 하는 소프트웨어를 다양한 관점에서 생각한 결과를 기술한다.

- Overview

'My LIittle Shoting Game'는 윈도우를 기반으로 하여 크로스 플랫폼을 목표로 하고 있다. 해당 게임은 포톤 서버를 통하여 플레이어 간 게임 환경을 동기화한다. 이 과정에서 여러 가지 문제를 직면할 것으로 예상된다.

- 서버와 플레이어의 동기화 시간 차이
- 서버에 너무 많은 정보의 동기화
- FPS의 장르 특성과 해당 게임에서 추구하는 게임의 그래픽

1) Problem #1

서버와 플레이어의 동기화 시간 차이로 인해 내 시점에서는 상대가 피해를 받았지만, 상대 시점에서는 그렇지 않은 경우가 발생할 수 있다.

2) Problem #2

서버에 어느 정보까지를 동기화 시킬 것인가에 대한 고민을 해야 합니다. 오브젝트의 체력정보와 같이 중요한 정보는 바로 동기화 해야하는 반면 총알 이펙트와 같이 게임 결과에 큰 영향이 없는 정보에 대해서는 동기화하게 되면 리소스의 낭비로 이어집니다.

3) Problem #3

FPS의 장르 특성상 복셀 그래픽을 활용하면 현실성이 필연적으로 떨어지는데 일인칭으로 진행되는 게임에서 현실성이 떨어지는 것은 게임 진행에 있어 불편할 수 있다.

NFRs

a. 게임엔진은 유니티를 사용한다.
b. 게임 동기화에 사용되는 시간을 60프레임 안으로 한다.
c. 동기화에 해당하는 내용은 게임 승패와 직결되는 정보에 한한다.

6. Glossary

용어	설명
Unity	게임 클라이언트를 제작하기 위한 기본적이 도구이다.
PhotonNetwork	멀티플레이를 위한 네트워크 서버이다.
Object	인 게임에서 존재하는 체력을 가진 개체이다.
복셀 그래픽	Volume + Pixel의 합성어로 '부피를 가진 픽셀'이라 할 수 있다. 작은 정육면체 또는 부피요소로 나눠질 수 있는 오브젝트를 렌더링하는 기술을 말한다.

7. References

1. Unity Documentation : https://docs.unity.com/
2. PhotonNetwork : https://www.photonengine.com/
3. 위키백과 - 오락 : https://ko.wikipedia.org/wiki/%EC%98%A4%EB%9D%BD