Network Game Programming

Term Project 추진 계획서



2012182039 정홍래

2013182043 최아도

2013182003 곽범식

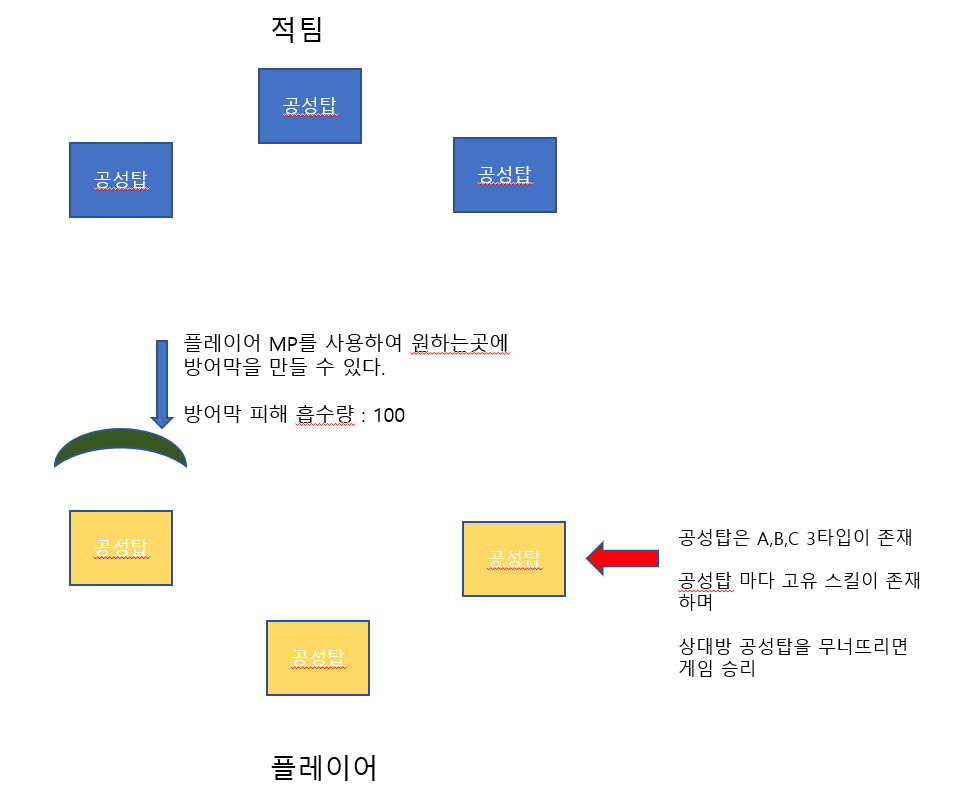
목차

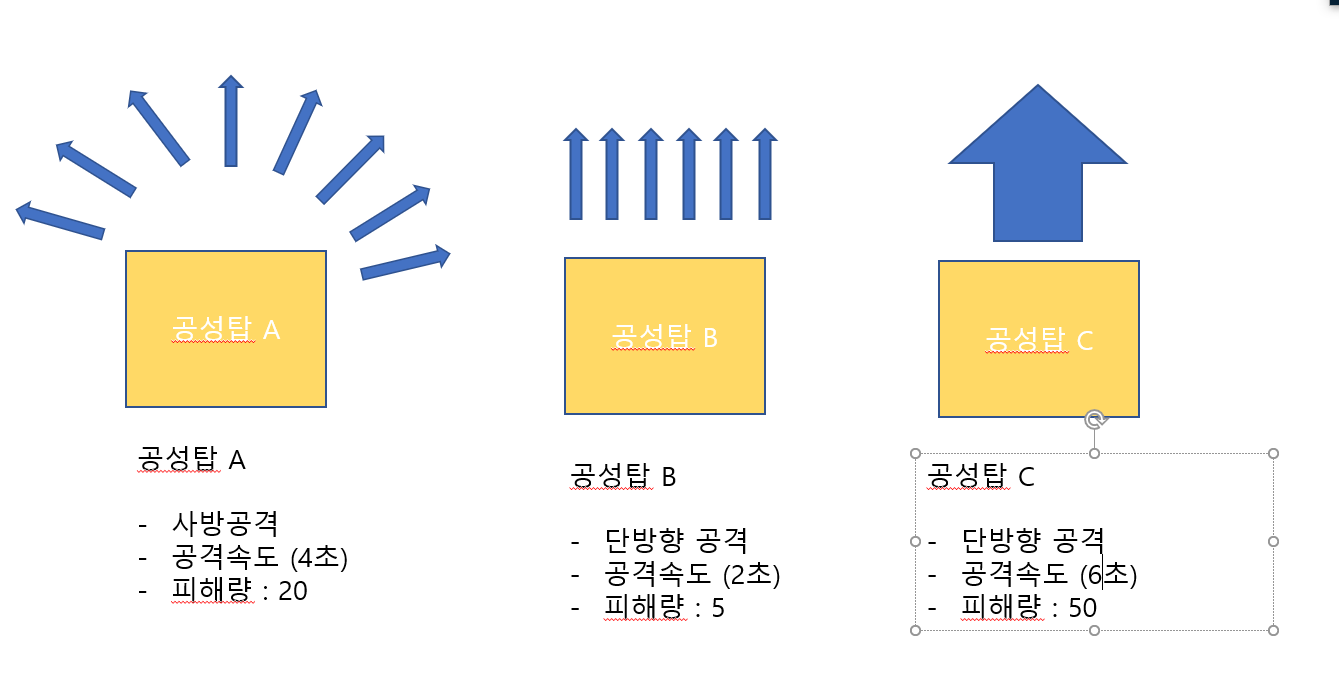
1. **게임 소개**
   1. 게임 제목2
   2. 게임 장르3
   3. 게임 설명2
   4. 게임 조작법3
   5. 게임 흐름 예상도2
2. **요구 사항 분석4**
   1. 기능 요구사항5
3. **High-level 디자인**
   1. 클라이언트 흐름도2
   2. 서버 흐름도3
   3. 스레드 동기화 방식2
4. **Low-level 디자인**
   1. 클라이언트 구조체, 함수2
   2. 서버 구조체, 함수3
5. **팀원 역할분담**
   1. 정홍래2
   2. 최아도3
   3. 곽범식2
6. **개발 환경**
7. **개발 일정**
8. **게임 소개**
   1. 게임 제목 - Dot War
   2. 게임 장르 - RTS
   3. 게임 설명

RTS장르의 게임으로 상대방 진형을 깨 부수어 승리를 거두는 방식이다. 플레이어는 자신의 진형에 3가지 종류의 공성탑을 지을 수 있으며 총 3개 지을 수 있다.

게임 시작 전에는 상대방의 공성탑 배치를 보지 못하지만, 게임이 시작하면 상대방 공성탑 배치를 확인하게 되며 게임을 진행한다.

상대방 공성탑 배치를 유의하며 자신의 공성탑에 적절한 스킬을 사용하여 상대방 모든 공성탑을 무너뜨리면 게임은 승리한다.





\*공성탑의 공격속도, 피해량, 체력(아직 미정)은 게임 난이도 조절하면서 수정할 가능성이 존재

* 1. 게임 조작법

<게임 시작 전>

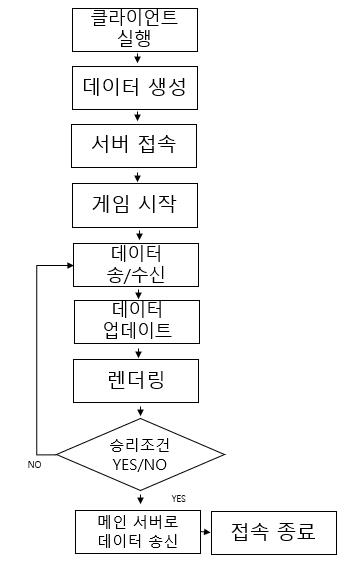
Space : 공성탑 배치

A키 : 공성탑 고르기

<게임 시작 후>

마우스 클릭 : 방어막 설치

* 1. 게임 흐름 예상도



1. **요구 사항 분석**
   1. 기능 요구 사항

* 네트워크 프로토콜은 TCP 서버를 사용한다.
* 관전 모드는 UDP 서버의 broadcast massage로 데이터를 전송 받는다. ( 확장기능 )
* 서버는 n개의 클라이언트를 수용한다.
* 대기방을 생성할 수 있다. (생성시 클라이언트가 간이 서버를 생성 : Host)
  + 대기방을 만들면 만든 대기방 정보 ( 방장의 IP주소, 포트번호 등 서버에 전송)
* 만들어진 대기방에 다른 클라이언트가 접속시 대기방 입장(connect)
  + 서버에서 만들어진 방의 목록을 받아와서 IP, 포트번호 로 연결
* 게임이 종료 될 경우 결과 서버에 저장 ( 승 / 패 )
  + But 연결이 강제로 끊겼을 경우, 끊어진 쪽의 패배로 기록할 수 있어야함.
* 서버는 멀티스레드를 이용한 이벤트 처리로 동기화 과정을 거친다.
  + 동시에 대기방에 입장하려는 클라이언트 간의 충돌이 일어날 수 있다.
* 방장은 상대편의 데이터를 recv 한 후 update 한 결과물을 다시 send한다.
  + 참가자는 받은 데이터를 기반으로 렌더링을 하며, 업데이트 하는 데이터를 보낸다.

딱 4개 1. Hp 2. 좌표 3. 유닛 종류 4. 유닛 상태 -> 구조체로 묶어라

배열, 리스트, 벡터 로 저장된걸 그냥 순차적으로 줘서 하나씩 그리면 된다.

클레시 로얄 ==> 서버(host) 전부 연산, 참가자 = > 조작만 (유닛 생성)

나머지 업데이트 => host가 전부 해줌

1. **High-level 디자인**
   1. 서버 흐름도
   2. 클라이언트 흐름도
   3. 스레드 동기화 방식
2. **Low-level 디자인**
   1. 클라이언트 구조체, 함수
   2. 서버 구조체, 함수
3. **팀원 역할분담**
   1. 정홍래
   2. 최아도
   3. 곽범식
4. **개발 환경**
5. **개발 일정**

|  |  |  |  |
| --- | --- | --- | --- |
| **날짜** | **정홍래** | **최아도** | **곽범식** |
| **11월 9일** |  |  |  |
| **11월 10일** |  |  |  |
| **11월 11일** |  |  |  |
| **11월 12일** |  |  |  |
| **11월 13일** |  |  |  |
| **11월 14일** |  |  |  |
| **11월 15일** |  |  |  |
| **11월 16일** |  |  |  |
| **11월 17일** |  |  |  |
| **11월 18일** |  |  |  |
| **11월 19일** |  |  |  |
| **11월 20일** |  |  |  |
| **11월 21일** |  |  |  |
| **11월 22일** |  |  |  |
| **11월 23일** |  |  |  |
| **11월 24일** |  |  |  |
| **11월 25일** |  |  |  |
| **11월 26일** |  |  |  |
| **11월 27일** |  |  |  |
| **11월 28일** |  |  |  |
| **11월 29일** |  |  |  |
| **11월 30일** |  |  |  |
| **12월 1일** |  |  |  |
| **12월 2일** |  |  |  |
| **12월 3일** |  |  |  |
| **12월 4일** |  |  |  |
| **12월 5일** |  |  |  |
| **12월 6일** |  |  |  |
| **12월 7일** |  |  |  |
| **12월 8일** |  |  |  |
| **12월 9일** |  |  |  |
| **12월 10일** |  |  |  |
| **12월 11일** |  |  |  |
| **12월 12일** |  |  |  |
| **12월 13일** |  |  |  |
| **12월 14일** |  |  |  |
| **12월 15일** |  |  |  |
| **12월 16일** |  |  |  |
| **12월 17일** |  |  |  |
| **12월 18일** | **최종 프로젝트 발표** | | |