|  |
| --- |
| **Report** |

**Team Meta.**

**조장** 심재경 2014010916

백형진 2014010911

박수빈 2014010908

김세훈 2014010904

* **INDEX** -

Report #1

* 1. **요구사항 명세서**
  2. Business Object ------------------------------------------------- p.3
  3. Use Case Diagram --------------------------------------------- p.3
  4. 기능적 요구사항 ------------------------------------------------ p.4

1.3.1 기능적 요구사항 목록 --------------------------------- p.4

1.3.2 Use Case 명세서 ----------------------------------------- p.4

* 1. 비기능적 요구사항 ---------------------------------------------- p.7

1.4.1 비기능적 요구사항 목록 ------------------------------- p.7

1.4.2 품질속성 시나리오 -------------------------------------- p.7

1.5 제약사항 ----------------------------------------------------------- p.10

* 1. **프로그램 계획서**

2.1 역할 분담 ------------------------------------------------------------- p.11

2.2 WBS -------------------------------------------------------------------- p.11

2.3 PERT Chart ----------------------------------------------------------- p.11

2.4 프로세스 적용 ------------------------------------------------------ p.12

2.5 참조 ------------------------------------------------------------------- p.14

|  |
| --- |
| **1. 요구사항 명세서** |

|  |
| --- |
| **1.1 Business Object** |

**1) 독특한 형식의 게임 컨트롤러**

- 이전에 없던 독특한 형식의 게임 컨트롤러를 개발하는 것이 목표이다.

**2) 개발한 컨트롤러의 맞춤게임**

- 컨트롤러의 특징을 이용한 맞춤형 게임을 개발하는 것이 목표이다.

|  |
| --- |
| **1.2 Use Case Diagram** |

|  |
| --- |
|  |

|  |
| --- |
| **1.3 기능적 요구사항** |

**1.3.1 기능적 요구사항 목록**

1) 컨트롤러 요구사항

|  |
| --- |
| - 진동감지기능 (두들기기)  - 회전각감지기능 (돌리기)  - 버튼입력감지기능 (누르기)  - 가속도감지기능 (흔들기)  - 통신기능 |

2) 게임 소프트웨어 요구사항

|  |
| --- |
| 1. 게임 제공 서비스  - 3D화면출력 서비스  - 랭킹 서비스  2. 캐릭터 기능  - 방향전환  - 점프  - 함정탈출  - 부스터 |

**1.3.2 Use Case 명세서**

**Use Case #1**

|  |
| --- |
| **▶ 유스케이스 명** : 방향전환  **▶ 엑터 명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임을 플레이할 때, 캐릭터의 방향을 전환한다.  **▶ 사전 조건** : 게임 플레이한다.  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 컨트롤러의 롤러를 돌린다.  2. 컨트롤러는 롤러 센서의 회전 각도를 측정한다.  3. 컨트롤러는 게임 소프트웨어로 방향전환 신호를 보낸다.  4. 게임 소프트웨어는 캐릭터의 방향을 전환한다.  - 대안 흐름  - 예외 흐름  1. 정상 흐름 3에서 측정된 회전각도가 15도이하일 경우, 방향전환 신호를 보내지 않는다. |

**Use Case #2**

|  |
| --- |
| **▶ 유스케이스명** : 함정탈출  **▶ 엑터명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자가 게임 플레이 도중 함정에 빠졌을 때, 캐릭터는 함정에서 탈출한다.  **▶ 사전 조건** : 게임 플레이한다.  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 컨트롤러를 흔든다.  2. 컨트롤러는 가속도 센서로 순간 가속도를 측정한다.  3. 컨트롤러는 게임 소프트웨어로 함정탈출 신호를 보낸다.  4. 게임 소프트웨어는 캐릭터를 함정에서 탈출시킨다.  - 대안 흐름  - 예외 흐름  1. 정상 흐름 3에서 측정된 순간 가속도가 25m/s2이하일 경우, 함정탈출 신호를 보내지  않는다. |

**Use Case #3**

|  |
| --- |
| **▶ 유스케이스명** : 달리기  **▶ 엑터명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임을 플레이할 때, 캐릭터를 달리게 한다.  **▶ 사전 조건** : 게임 플레이한다.  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 컨트롤러를 두드린다.  2. 컨트롤러는 진동 감지 센서로 두드림을 감지한다.  3. 컨트롤러는 게임 소프트웨어로 달리기 신호를 보낸다.  4. 게임 소프트웨어는 캐릭터를 달리게 한다.  - 대안 흐름  - 예외 흐름 |

**Use Case #4**

|  |
| --- |
| **▶ 유스케이스명** : 점프  **▶ 엑터명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임을 플레이할 때, 케릭터를 점프시킨다.  **▶ 사전 조건** : 게임 플레이한다.  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 컨트롤러의 버튼을 누른다.  2. 컨트롤러는 버튼 누름을 감지한다.  3. 컨트롤러는 게임 소프트웨어로 점프 신호를 보낸다.  4. 게임 소프트웨어는 캐릭터를 점프시킨다.  - 대안 흐름  - 예외 흐름 |

**Use Case #5**

|  |
| --- |
| **▶ 유스케이스명** : 랭킹확인  **▶ 엑터명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임 클리어 타임 기준으로 측정된 랭킹을 확인한다.  **▶ 사전 조건** : 해당 없음  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 게임을 실행한다.  2. 게임 소프트웨어는 메뉴를 출력한다.  3. 사용자는 메뉴 선택에서 랭킹확인을 누른다.  4. 게임 소프트웨어는 저장된 클리어 타임 랭킹을 보여준다.  - 대안 흐름  1. 정상 흐름 3에서 “게임시작” 를 선택할 경우, 본 게임이 시작된다.  2. 정상 흐름 3에서 “게임종료” 를 선택할 경우, 게임이 종료된다.  - 예외 흐름 |

**Use Case #6**

|  |
| --- |
| **▶ 유스케이스명** : 랭킹등록  **▶ 엑터명** : User1, User2  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임을 클리어한 후, 클리어 타임으로 랭킹등록이 가능하다.  **▶ 사전 조건** : 해당 없음  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 게임을 클리어한다.  2. 게임 소프트웨어는 측정된 클리어 타임이 출력한다.  3. 게임 소프트웨어는 랭킹 등록 여부를 묻는다.  4. 게임 소프트웨어는 사용자가 YES라고 누를 경우, 랭킹 등록을 한다.  5. 게임 소프트웨어는 저장된 랭킹을 보여준다.  - 대안 흐름  1. 정상 흐름 4에서 “등록하지 않는다” 를 선택할 경우, 메뉴 화면으로 돌아간다.  - 예외 흐름  1. 정상 흐름 1에서 사용자가 클리어 실패했을 경우, 클리어 실패 문구를 출력한다. |

**Use Case #7**

|  |
| --- |
| **▶ 유스케이스명** : 클리어타임 측정  **▶ 엑터명** : Timer  **▶ 유스케이스 개요 및 설명**  - 게임 소프트웨어는 사용자가 게임을 하는 동안 시간을 측정하고 있다.  **▶ 사전 조건** : 해당 없음  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 게임을 플레이한다.  2. 게임 소프트웨어는 게임 시작 시간을 저장한다.  3. 사용자는 클리어한다.  4. 게임 소프트웨어는 게임 종료 시간과 시작 시간 차이를 계산하여 클리어 타임을 측정한다.  - 대안 흐름  - 예외 흐름  1. 정상 흐름 3에서 사용자가 클리어 실패했을 경우, 클리어 실패 문구를 출력한다. |

**Use Case #8**

|  |
| --- |
| **▶ 유스케이스명** : 게임플레이  **▶ 엑터명** : X  **▶ 유스케이스 개요 및 설명**  - 사용자는 게임 소프트웨어를 시작 시킨 후, 게임 시작 메뉴를 선택한다.  **▶ 사전 조건** : 해당 없음  **▶ 작업 흐름**  - 정상흐름  1. 사용자는 게임 소프트웨어를 실행한다.  2. 게임 소프트웨어는 메뉴 선택 화면을 출력한다.  3. 사용자는 메뉴 중 게임 시작 메뉴를 선택한다.  - 대안 흐름  1. 정상 흐름 3에서 ‘랭킹확인’을 선택할 경우, 랭킹을 출력한다.  2. 정상 흐름 3에서 ‘종료’를 선택할 경우, 게임 소프트웨어는 종료된다.  - 예외 흐름  1. 정상 흐름 1에서 컨트롤러가 연결되지 않을 경우, 경고 문구가 출력된다. |

|  |
| --- |
| **1.4비기능적 요구사항** |

**1.4.1 비기능적 요구사항 목록**

1) 컨트롤러 비기능적

|  |
| --- |
| - 컨트롤러와 PC와의 통신 응답시간이 빨라야 한다.  - 평균적인 손 크기에 맞는 컨트롤러 크기를 정해야 한다 |

2) 게임 소프트웨어

|  |
| --- |
| - 랭킹조작을 할 수 없도록 암호화 하여야 한다.  - 게임은 컨트롤러가 인식된 상태에서만 동작한다. |

**1.4.2 품질속성 시나리오**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **품질속성** | **속성 상세화** | **시나리오** | **우선순위** | |
| **중요성** | **난이도** |
| 성능 | 연결 응답 최소화 | Sc1. 사용자가 컨트롤러에 입력을 했을 시, 게임은 실시간으로 0.05초 이내 반응해야 한다. | H | M |
| 성능 | 센서 반응 수치 설정 | Sc2. 사용자가 컨트롤러에 입력을 했을 시, 컨트롤러의 센서는 정확하게 반응(두드리기, 돌리기, 누르기, 흔들기)해야 한다.  이러한 반응은 연속 두드리기 감지 간격이 0.2초이내, 돌렸을 때, 15도이상돌려야 감지한다. 흔들기는 25m/s2여야한다. | H | M |
| 테스팅 | 모듈별 테스트 | Sc3. 개발자가 하드웨어, 통신, PC별로 단위 테스트 수행한다.  하드웨어는 센서별로, 통신는 컨트롤러 인식, PC는 컨틑롤러 입력 확인 및 게임 적용이 되는지 확인한다. | H | H |

1) 성능 시나리오

- 사용자가 컨트롤러에 입력을 했을 시, 게임은 실시간으로 반응해야 한다.

이러한 반응은 평균 0.05초이내여야 한다.

|  |  |
| --- | --- |
|  | QAS\_1 |
| 환경 | 게임 실행 중 |
| 소스 | 사용자 |
| 자극 | 컨트롤러 입력에 대한 게임의 반응  - 돌리기  - 흔들기  - 두드리기  - 누르기 |
| 대상 | 게임 |
| 응답 | 캐릭터가 움직인다.  - 돌리기 -> 방향전환  - 흔들기 -> 함정탈출  - 두드리기 -> 달리기  - 누르기 -> 점프 |
| 응답척도 | 0.05초 이내 |

- 사용자가 컨트롤러에 입력을 했을 시, 컨트롤러의 센서는 정확하게 반응(두드리기, 돌리기,

누르기, 흔들기)해야 한다. 이러한 반응은 연속 두드리기 감지 간격이 0.2초이내, 돌렸을 때,   
15도이상 돌려야 감지한다. 흔들기는 25m/s2여야한다.

|  |  |
| --- | --- |
|  | QAS\_2 |
| 환경 | 게임 실행 중 |
| 소스 | 사용자 |
| 자극 | 컨트롤러를 사용한다. 그 기능으로는 다음과 같다.  -두드리기  -돌리기  -누르기  -흔들기 |
| 대상 | 게임 컨트롤러 |
| 응답 | 컨트롤러 기능에 대응하는 센서의 반응 |
| 응답척도 | 각 기능에 대한 정확도는 다음과 같다.  -두드리기 : 연속 입력에 대한 반응 감지 간격은 0.2초 이내  -돌리기 : 15도이상 돌렸을 때, 반응 감지  -누르기 : 실시간 반응 감지  -흔들기 : 순간 가속도 25m/s2 이상일 때, 반응 감지 |

2) 변경용이성 시나리오

- 개발자가 게임의 버그를 발견했을 때, 버그 수정하려고 한다.

이에 따라 개발자는 점검 시점에 코드를 수정할 것이다. 이것은 24시간이내 이루어져야 한다.

이 행위가 버그 이외의 다른 시스템에 영향을 주면 안된다.

|  |  |
| --- | --- |
|  | QAS\_3 |
| 환경 | 버그 발견 후, 코드 수정하는 시기 |
| 소스 | 개발자 |
| 자극 | 버그 수정 |
| 대상 | 게임 코드 |
| 응답 | 버그 이외 다른 시스템에 영향을 주면 안된다. |
| 응답척도 | 24시간 이내 수정 완료해야 한다. |

3) 시험 용이성 시나리오

- 개발자가 하드웨어, 통신, PC별로 단위 테스트 수행한다.

하드웨어는 센서별로, 통신는 컨트롤러 인식, PC는 컨틑롤러 입력 확인 및 게임 적용

|  |  |
| --- | --- |
|  | QAS\_4 |
| 환경 | 테스트 기간 |
| 소스 | 개발자 |
| 자극 | 테스트 수행 |
| 대상 | 게임 컨트롤러, 컨트롤러와 PC와의 통신, 게임 |
| 응답 | 게임 컨트롤러의 조작에 대한 게임의 반응을 확인  - 돌리기  - 흔들기  - 두드리기  - 누르기 |
| 응답척도 | 각 센서에 대한 테스트케이스의 척도는 다음과 같다.  - 돌리기 : 10/10  - 흔들기 : 7/10  - 두드리기 : 7/10  - 누르기 : 10/10 |

4) 사용편의성 시나리오

- 정상적인 게임 플레이 중, 컨트롤러와 PC의 연결이 끊어졌다. 게임은 일시 중지되고,   
키보드 조작이 가능하도록 한다.

|  |  |
| --- | --- |
|  | QAS\_5 |
| 환경 | 게임 실행 중 |
| 소스 | 게임 컨트롤러 |
| 자극 | 연결 끊어짐 |
| 대상 | PC와 게임 |
| 응답 | 게임 일시 정지를 한 후, 키보드 조작 활성 |
| 응답척도 | 키보드로 종료할 수 있다. |

|  |
| --- |
| **1.5 제약사항** |

|  |
| --- |
| 1) 컨트롤러와 PC의 연결은 유선연결로 한다.  2) 전용게임에서만 컨트롤러가 작동한다.  3) 진동 센서, 자이로 센서, 롤러 센서, 버튼을 사용한다. |

|  |
| --- |
| **2. 프로그램 계획서** |

|  |
| --- |
| **2.1 역할분담** |

**TL 심재경** : 팀을 이끌 리더로서 작업 분담과 기술적 사항을 해결합니다.

**PM 박수빈** : 팀의 프로세스를 관리하는 책임자로서 프로세스 적용 방법을 가이드 합니다.

**PRM 김세훈** : 팀의 계획을 관리하는 책임자로서 계획 수립 및 위험감시를 합니다.

**SM 백형진** : 팀의 결과물을 관리하는 책임자로서 결과물 관리 및 진행상황을 감시합니다.

|  |
| --- |
| **2.2 WBS (Work Breakdown Structure)** |

|  |
| --- |
|  |

|  |
| --- |
| **2.3 PERT Chart** |

|  |
| --- |
| EMB0000183c3351 |

<그림 p.14 참조>

|  |
| --- |
| **2.4 프로세스 적용** |

RUP과 XP의 장단점 분석

|  |  |  |
| --- | --- | --- |
| 프로세스 | RUP | XP |
| 장점 | - 사용자의 요구사항을 충족시키는 시스템을 구축하다.  - 사용자들의 요구사항의 관리가 가능하다.  - 지속적으로 소프트웨어의  품질을 보장 가능하다.  - 시각적 모델을 가지는 소프트웨어의 개발이 가능해진다. | -저비용 소규모 프로젝트에 상대적으로 유리하다.  - 개발자 사이의 의사소통이 원할 하다.  - 여러 가지 잦은 요구사항에 대처하기 쉽다.  - 모든 일을 단순하게 처리가 가능하다.  - 짧은 주기로 “ 개발 - 테스트 - 배포”가 가능하다. |
| 단점 | - 범용적에 적합한 모델이기에 특정업무에 대해서는 적용이 어렵다.  - XP에 비해 산출물이 많이 나온다.  - 무겁고, 세부적, 반복중심이다.  - 전체의 소프트웨어 과정이 무조건 개발과정 이여야 한다.  - Multi-system기반의 개발환경을 지원하지 않는다.  - 도구중심의 접근방식은 복잡한 시스템에서는 적절하지 않다. | - 대규모 프로젝트에는 상대적으로 불리하다.  - 전체적인 설계단계가 없다.  - 좋은 설계를 위해 비용이 증가한다.  - 전체적으로 요구사항이 많다.  - 중요도는 주관적이고, 전반적인 설계에 의해서 Up-fornt design 이라고 불린다.  - 팀워크 의사소통 우선순위가 강조되지만 동적인 환경 부담이 제기된다. |

선정 프로세스 : XP

선정 이유 :

1) 4명으로 이루어진 소규모 프로젝트이기 때문이다.

2) 팀원들 사이에 의사소통이 원활하고, 같은 환경에서 생활하기 때문에 동적인 환경 부담이 줄어든다.

3) 실질적인 개발기간이 약 한 달 정도이기 때문에 짧은 주기로 “ 개발 - 테스트 - 배포”가 가능한 XP로 한다.

|  |
| --- |
| **2.5 참조** |
|  |

<PERT Chart>

