목차

[1 Preface 4](#_Toc514346631)

[1.1 Objective 4](#_Toc514346632)

[1.2 Readership 4](#_Toc514346633)

[1.3 Document Structure 4](#_Toc514346634)

[1.4 Version of the Document 5](#_Toc514346635)

[2 Introduction 6](#_Toc514346636)

[2.1 Objectives 6](#_Toc514346637)

[2.2 Applied Diagram 6](#_Toc514346638)

[2.3 Applied Tool 7](#_Toc514346639)

[3 System Architecture 8](#_Toc514346640)

[3.1 Objectives 8](#_Toc514346641)

[3.2 System Organization 8](#_Toc514346642)

[3.3 Package Diagram 10](#_Toc514346643)

[3.4 Deployment Diagram 10](#_Toc514346644)

[4 Start & End System 10](#_Toc514346645)

[4.3 Sequence Diagram 10](#_Toc514346646)

[4.4 State Diagram 12](#_Toc514346647)

[5 Question & Answer System 13](#_Toc514346648)

[5.1 Objectives 13](#_Toc514346649)

[5.2 Class Diagram 13](#_Toc514346650)

[A. QnAHandler 13](#_Toc514346651)

[A.1 Attributes 13](#_Toc514346652)

[A.2 Methods 13](#_Toc514346653)

[B. DialogflowManager 13](#_Toc514346654)

[A.1 Attributes 13](#_Toc514346655)

[A.2 Methods 13](#_Toc514346656)

[C. Data 13](#_Toc514346657)

[A.1 Attributes 14](#_Toc514346658)

[A.2 Methods 14](#_Toc514346659)

[5.3 Sequence Diagram 14](#_Toc514346660)

[A. Question Analysis 14](#_Toc514346661)

[5.4 State Diagram 15](#_Toc514346662)

[A. Question Analysis 15](#_Toc514346663)

[B. Answer Distinguishment 16](#_Toc514346664)

[C. Answer Comparison 17](#_Toc514346665)

[6 Elimination System 17](#_Toc514346666)

[6.1 Objectives 17](#_Toc514346667)

[6.2 Class Diagram 17](#_Toc514346668)

[A.2 Methods 18](#_Toc514346669)

[A.1 Attributes 18](#_Toc514346670)

[A.2 Methods 18](#_Toc514346671)

[6.3 Sequence Diagram 18](#_Toc514346672)

[6.4 State Diagram 19](#_Toc514346673)

[7 Help & log System 19](#_Toc514346674)

[7.1 Objectives 19](#_Toc514346675)

[7.2 Class Diagram 19](#_Toc514346676)

[7.3 Sequence Diagram 19](#_Toc514346677)

[7.4 State Diagram 19](#_Toc514346678)

[8 DB Management System 19](#_Toc514346679)

[8.1 Objectives 19](#_Toc514346680)

[8.2 Class Diagram 19](#_Toc514346681)

[A.1 Attributes 20](#_Toc514346682)

[해당없음 20](#_Toc514346683)

[8.3 Sequence Diagram 20](#_Toc514346684)

[8.4 State Diagram 21](#_Toc514346685)

[9 Protocol Design 22](#_Toc514346686)

[9.1 Objectives 22](#_Toc514346687)

[9.2 JSON 22](#_Toc514346688)

[9.3 Protocol Description 22](#_Toc514346689)

[10 Database Design 22](#_Toc514346690)

[10.1 Objectives 22](#_Toc514346691)

[10.2 ER Diagram 22](#_Toc514346692)

[10.3 Relation Schema 22](#_Toc514346693)

[11 Testing Plan 22](#_Toc514346694)

[11.1 Objectives 22](#_Toc514346695)

[11.2 Testing Policy 22](#_Toc514346696)

[11.3 Test Case 23](#_Toc514346697)

[12 Development Environment 23](#_Toc514346698)

[12.1 Objectives 23](#_Toc514346699)

[12.2 Programming Language & IDE 23](#_Toc514346700)

[12.3 Coding Rule 23](#_Toc514346701)

[12.4 Version Management Tool 23](#_Toc514346702)

[13 Develop Plan 23](#_Toc514346703)

[13.1 Objectives 23](#_Toc514346704)

[13.2 23](#_Toc514346705)

[14 Index 23](#_Toc514346706)

[14.1 Table Index 23](#_Toc514346707)

[14.2 Figure Index 23](#_Toc514346708)

[14.3 Diagram Index 23](#_Toc514346709)

[15 Reference 23](#_Toc514346710)

1. Preface
   1. Objective

Preface에서는 본 문서의 독자를 정의하고, 구조를 소개한다. 각 구조는 목적을 먼저 서술한다. 또한, 문서의 버전을 서술한다.

* 1. Readership

본 문서의 독자는 다음과 같다. software engineers, architects, 그리고 이와 관련된 팀원 및 외부 사람들을 독자로 정의한다.

* 1. Document Structure
     1. Preface

Preface에서는 본 문서의 독자를 정의하고, 구조를 소개한다. 각 구조는 목적을 먼저 서술한다. 또한, 문서의 버전을 서술한다

* + 1. Introduction

이 목차에서는 본 문서에서의 System design에 대해 사용된 소프트웨어(tools같은)과 다이어그램들을 서술한다.

* + 1. System Architecture

System Architecture에서는 현재 개발하고자 하는 시스템의 전반에 대해 서술한다. 즉, 전체적인 시스템 구조를 Block Diagram, Package diagram, Deployment diagram을 이용하여 나타내고, 아래의 목차 순으로 설명한다.

* + 1. Start / End System

게임 안의 플레이어 디스코드 안에서 봇을 초대하고, 게임을 시작하여 발생하는 과정에 대해서 class diagram, sequence diagram, state diagram을 통해 설명한다.

* + 1. Question & Answer System
    2. Elimination System
    3. Help & log System
    4. Protocol Design

Protocol Design에서는 서브시스템들이 상호작용하는 프로토콜에 대해 서술한다.

* + 1. Database Design

DB Design은 사용된 DB의 구조 정의를 서술한다.

* + 1. Testing Plan

각 서브 시스템이 예상된 실행을 보여주는지를 테스팅하여, 결함을 찾기 위한 testing plan을 서술한다.

* + 1. Development Environment

시스템을 개발할 때에 필요한 것들을 서술한다. programming language, IDE 등을 서술한다.

* + 1. Develop Plan

개발 계획에 대해 서술한다.

* + 1. Index

Index는 본 문서에서 보여준 Figure, table, diagram들의 순서를 표시하여 각 index의 내용이 본 문서 안에서의 위치를 알 수 있다.

* 1. Version of the Document
     1. Version Format

version 번호는 Major와 Minor number(format : {major, minor)의 형태로 표현한다. 본 문서의 버전은 0.1부터 시작한다.

* + 1. Version Management Policy

설계명세서를 수정할 때 마다 버전을 업데이트한다. 다만, 변경 시간 차가 짧을 경우 버전 번호를 업데이트하지 않고, 하나의 버전으로 취급한다. 이미 완성된 부분을 변경할 시, Minor number를 증가시키며, 새로운 부분을 추가 및 문서의 구성 변경 시 Major number를 변경한다.

* + 1. Version Update History

|  |  |
| --- | --- |
| Version | Explanation |
| 0.1 | 설계 명세서 문서의 목차 생성 |
| 1.0 | Preface와 Introduction 부분 작성 |
| 2.0 | Start & End System 작성. |
|  |  |
|  |  |
|  |  |

1. Introduction
   1. Objectives

Introduction은 본 문서에서 기술되어질 system Design을 위해 사용된 diagram 및 tool들에 대해 서술한다.

* 1. Applied Diagram
     1. UML

[Figure 첨부 바람]

United Modeling Language(UML)은 실용적인 여러 종류의 다이어그램들을 통합하여 만들어진 모델링 언어로 object-oriented software 개발할 때 산출 물을 명세화, 시각화, 문서화할 때 사용한다. 표기 방법을 사용하여 개발하고자 하는 소프트웨어의 구조적 청사진을 시각화하여 의사소통의 도구로 사용되어진다.

* + - 1. Package Diagram

개발할 소프트웨어을 이해하기 위한 목적으로 abstract 한 component들을 그룹 단위로 organization하기 위한 메커니즘

이다. 이 매커니즘을 통해 전체 시스템의 간략한 이해를 도와 준다. package안 의 component로는 package, class, interface, object 등을 포함할 수 있다.

* + - 1. Deployment Diagram
      2. Class Diagram
      3. State Diagram
      4. Sequence Diagram
    1. ER Diagram

데이터베이스에서 각 개체들(Entities)의 관계를 표현하기 위해 사용하는 다이어그램으로, DB에 저장될 데이터의 형식 정의와 제약조건(Constraints)들의 정의하기 위해서 사용된다.

[Figure 첨부 필요?]

* 1. Applied Tool
     1. Flow Chart Maker & Online Diagram Software

본 문서에서 시스템 설계와 관련된 다이어그램들의 작성은 'Flow Chart Maker & Online Diagram Software' tool을 사용하였는데, 이 tool은 다양한 UML에 말고도 다양한 diagram들을 지원하고, 컴퓨터 내에 설치 회원가입을 하지 않고도 online 상에서 이용할 수 있기 때문에 사용되어졌다.

[ Figure ]

* + 1. Amazon Web Services(AWS)

* + 1. DialogFlow

* + 1. Pycharm

본 프로젝트 과정에서 사용될 개발 도구이다. 본 프로젝트의 프로그래밍 언어로 python을 이용할 예정이기에 이 도구를 사용하게 되었다.

[ figure ]

* + 1. Discord

Discord는 게이밍 공동체를 위해 개발된 VoIP 응용 소프트웨어이다. 여러 OS에서도 별 문제 없이 이용되어지고, 이 소프트웨어는 메신저로의 기능, 정보 공유, 관리 기능 등을 제공하고 있어 여러 게임 유저들이 이용하고 있다.

1. System Architecture
   1. Objectives

System Architecture에서는 현재 우리 팀에서 개발하고자 하는 시스템에 대해 전반적인 구조를 서술한다. 이 서술 과정에서 Block Diagram으로 나타내고, 실제 어떻게 사용되는 지를 Package diagram과 deployment diagram으로 사용하여 설명한다.

* 1. System Organization

우리의 봇 시스템은 Client-server Model을 사용하여 구현된다.

[전체적인 시스템의 대한 설명이 필요할 거같음]

* + 1. Start & End System

Start & End System의 경우, 간단히 말하면, discord 상에서 봇을 초대한 후에 게임의 시작을 위한 초기화 기능, 게임의 end 조건을 충족하여, 그 결과에 대해 알리고 게임 과정에서 생성된 data information 정보를 DB에 저장하거나 갱신 요청하고 게임에 사용된 임시 데이터를 제거하는데 기능을 갖는다.

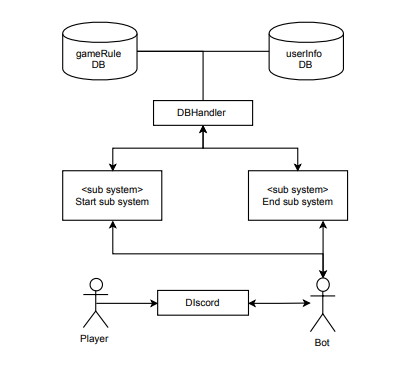
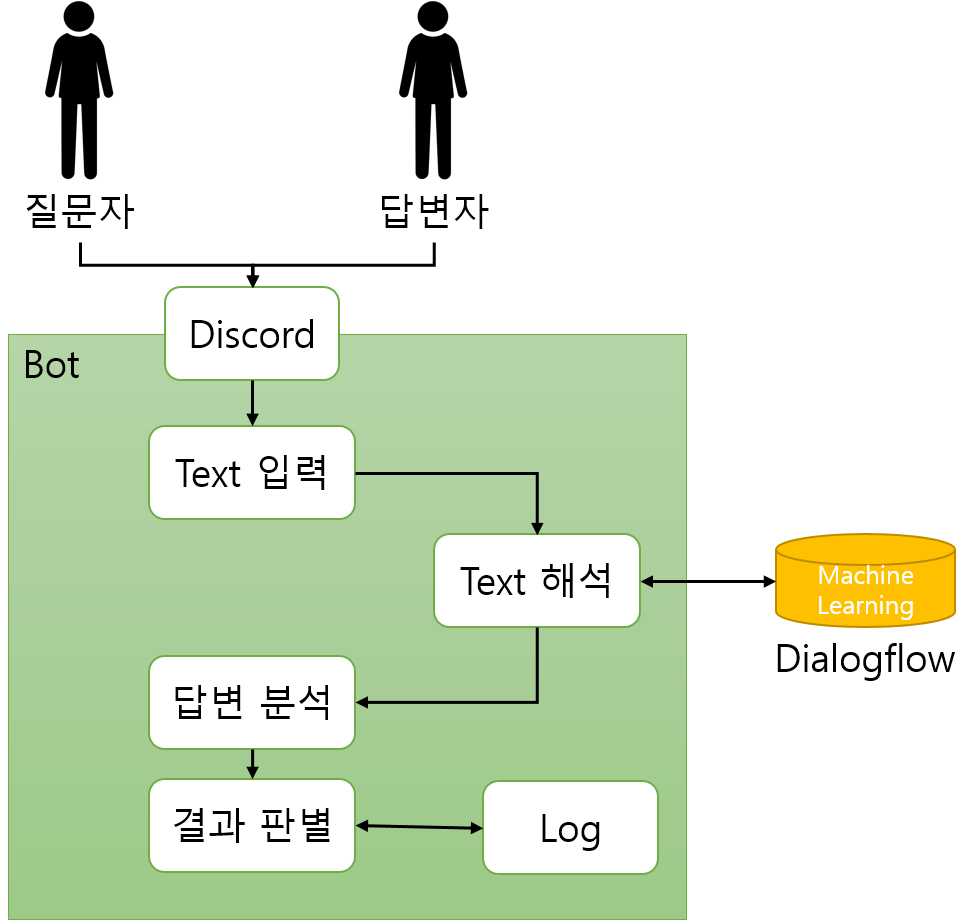


Diagram Start & End system

* + 1. Question & Answer System

Q&A System은 질문자와 답변자 두 종류의 참여자가 Discord상에 입력한 text를 각각 질문과 답변으로 해석하고,답변자가 올바른 답변을 하였는지 분석하여 결과를 내는 시스템이다.참여자와 Bot 사이를 Discord가 중개하고 있으며, Bot은 자연어의 해석 작업을 위해 Dialogflow API를 사용한다.



Diagram

* + 1. Elimination System
    2. Help & log System
    3. DB Management System
  1. Package Diagram
  2. Deployment Diagram

1. Start & End System
   1. Objectives

User가 Discord 상에서 우리의 봇을 초대하고 게임을 하기 위해 봇(game Handler)를 통해 initialization 기능을 통하여 게임을 시작을 진해아고, 게임이 끝나기 위한 조건이 충족하여 받은 정보를 토대로 게임을 종료하기 위한 기능들에 대한 설계를 설명한다. class diagram, sequence diagram, state diagram을 통해 Start & End system의 구조를 표현하고 설명한다.

* 1. Class Diagram
  2. Sequence Diagram

여기서 Bot actor는 handler이다.

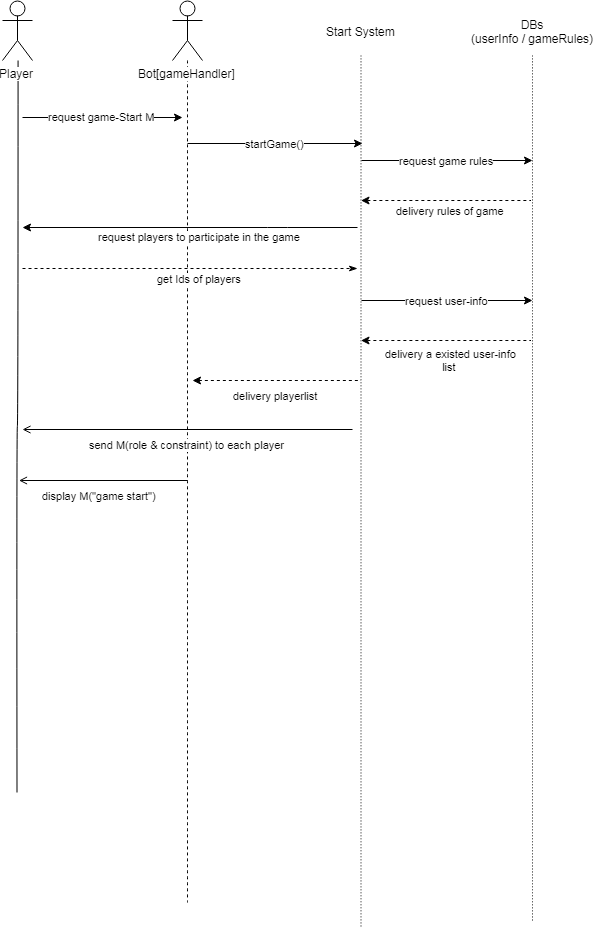


Diagram Start Sub-System sequence diagram

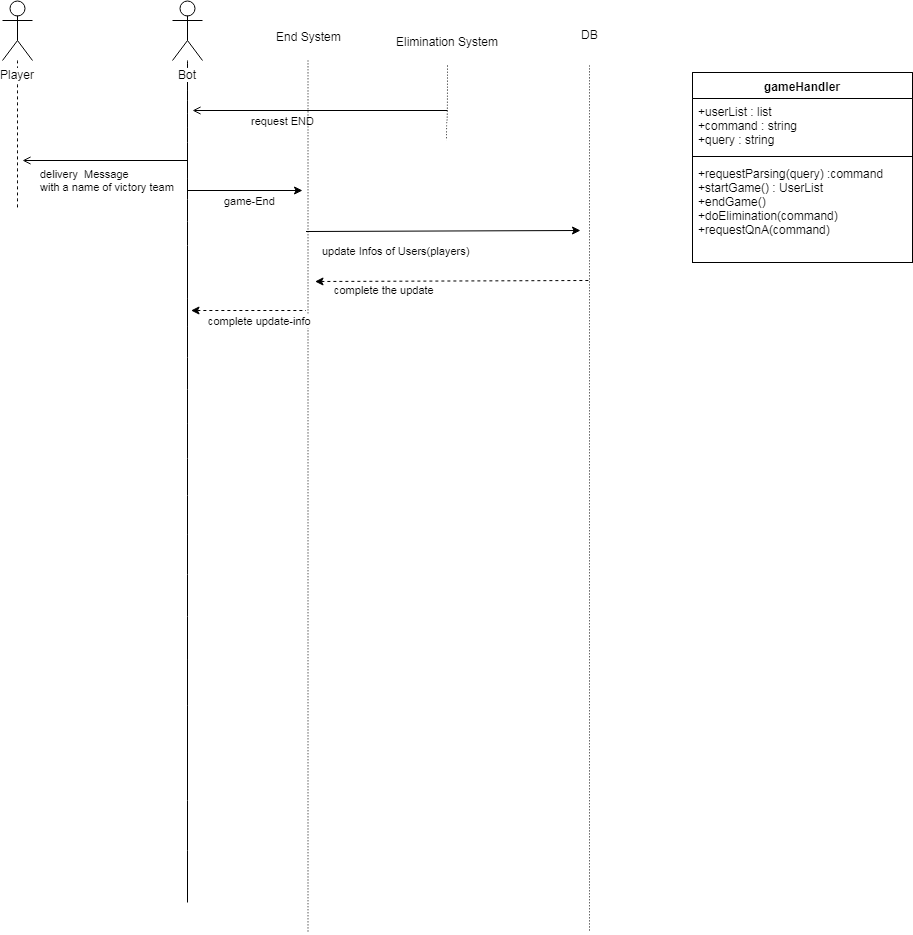


Diagram End sub-System sequence diagram

* 1. State Diagram

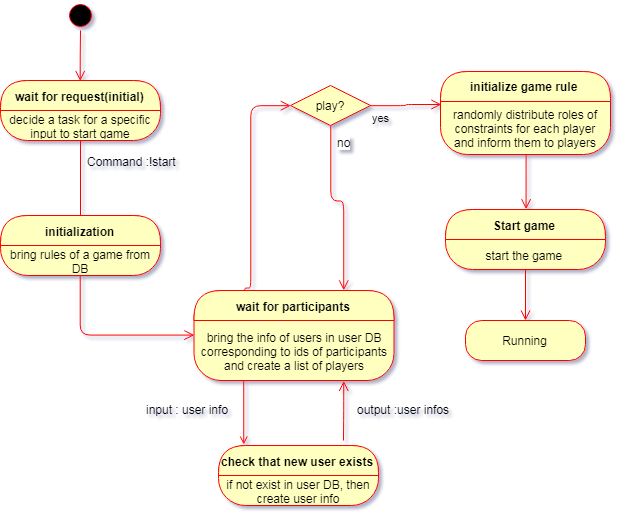


Diagram Start Sub-system State Diagram

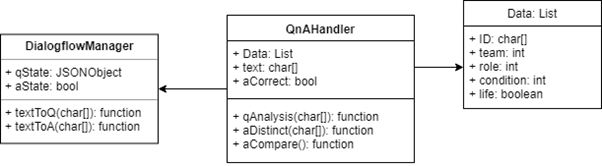
C:\Users\hoo\Desktop\설계명세서관련\endState.png

Diagram End Sub-system State Diagram

1. Question & Answer System
   1. Objectives

참여자가 Discord 상에 입력한 질문과 답변들을 해석하고 처리해주는 시스템을 설명한다. Q&A System의 구조를 표현하기 위해 Class Diagram, Sequence Diagram, State Diagram을 사용한다.

* 1. Class Diagram



* + 1. QnAHandler
       1. Attributes

+Data: 참여자들의 ID, 팀,역할,제약조건 정보

+text: 참여자가 Discord를 통해 입력한 문장

+aCorrect: 답변자가 해야하는 올바른 대답

* + - 1. Methods

+qAnalysis(char[]): 질문 text를 JSON 형태로 번역한 뒤,답변자의 개인정보를 토대로 올바른 대답을 계산한다.

+aDistinct(char[]): 답변 text를 Boolean형태로 번역한다.

+aCompare(): 올바른 대답과 실제 대답을 비교하고, 결과에 따라서 EliminationHandler에 접근한다.

* + 1. DialogflowManager
       1. Attributes

+qState: 질문 text를 JSON 형태로 번역한 것

+aState: 답변 text를 Boolean 형태로 번역한 것

* + - 1. Methods

+textToQ(char[]): 질문 text를 Dialogflow를 통해서 JSON 형태로 번역한다.

+textToA(char[]): 답변 text를 Dialogflow를 통해서 Boolean 형태로 번역한다.

* + 1. Data
       1. Attributes

+ID : 참여자들을 지칭하는 식별자

+team : 해당 참여자가 소속된 팀

+role : 해당 참여자가 부여 받은 역할

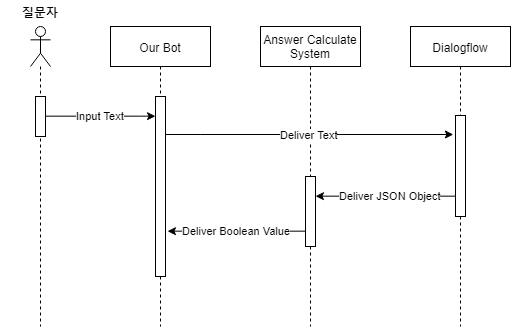
+condition : 해당 참여자가 부여 받은 제약조건

+life : 해당 참여자가 게임 내에서 생존 중인지를 나타낸다.

* + - 1. Methods

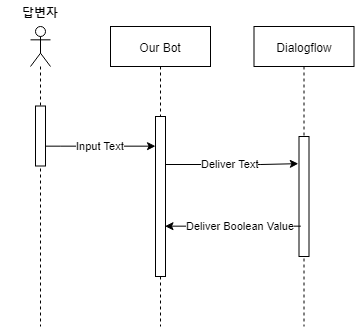
해당 없음

* 1. Sequence Diagram
     1. Question Analysis



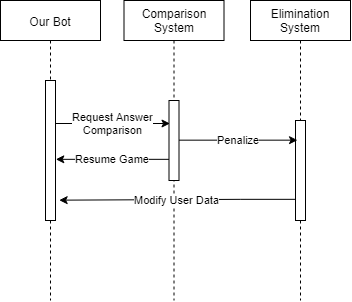
Diagram

* + 1. Answer Distinguishment



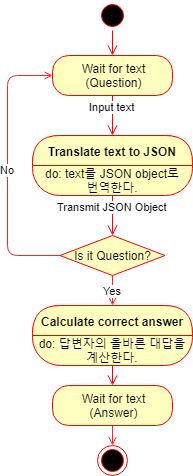
Diagram

* + 1. Answer Comparison



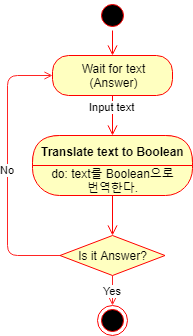
Diagram

* 1. State Diagram
     1. Question Analysis



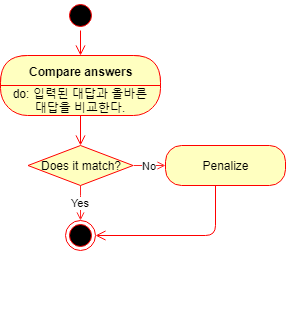
Diagram

* + 1. Answer Distinguishment



Diagram

* + 1. Answer Comparison

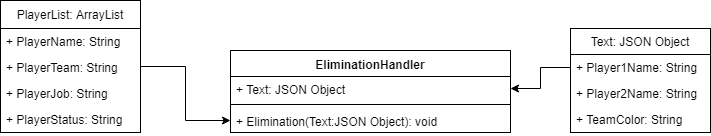


Diagram

1. Elimination System
   1. Objectives

한 플레이어가 특정 플레이어를 상대 팀의 왕으로 지목했을 경우 이를 처리해주는 시스템의 설계를 설명한다. Class Diagram, Sequence Diagram, State Diagram을 통해 본 시스템의 구조를 설명한다

* 1. Class Diagram



Diagram

* + 1. EliminationHandler
       1. Attributes

Text: 누가 누굴 지목했는지에 대한 정보

* + - 1. Methods

Elimination(Text): Text에 담긴 정보를 토대로 Elimination 과정을 진행한다.

* + 1. Test : JSON Object
       1. Attributes

Player1Name : 지목한 플레이어의 이름

Player2Name : 지목당한 플레이어의 이름

TeamColor : 지목한 플레이어의 소속 팀

* + - 1. Methods

해당없음

* + 1. PlayerList: ArrayList
       1. Attributes

PlayerName: 플레이어의 이름

PlayerTeam: 플레이어의 소속 팀

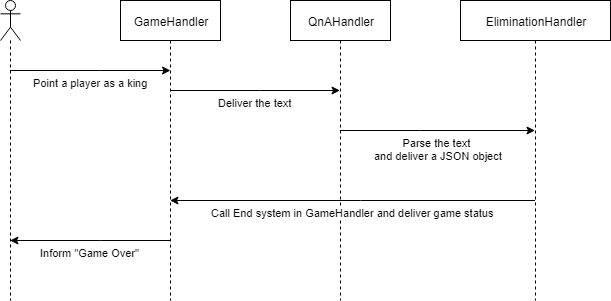
PlayerJob: 플레이어의 게임 내 지위

PlayerStatus: 플레이어의 생존 여부

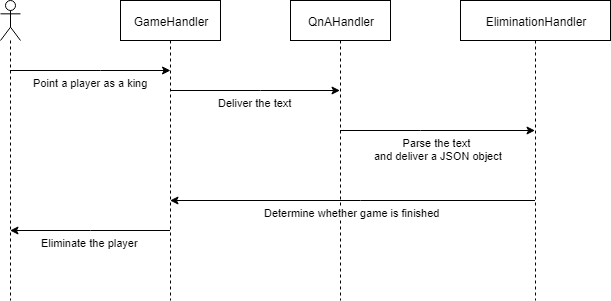
* + - 1. Methods

해당없음.

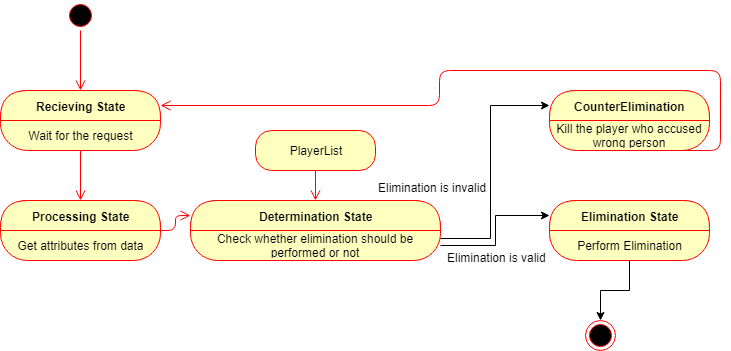
* 1. Sequence Diagram
     1. “Valid” Elimination: the player accused right person



* + 1. “Invalid” Elimination: the player accused wrong person.



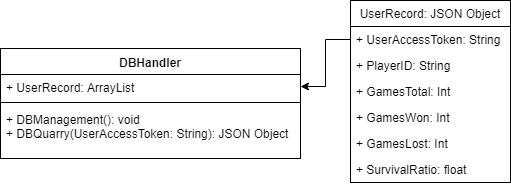
* 1. State Diagram



1. Help & log System
   1. Objectives
   2. Class Diagram
   3. Sequence Diagram
   4. State Diagram
2. DB Management System
   1. Objectives

본 봇의 Database를 관리해주는 시스템의 설계를 설명한다. Class Diagram, Sequence Diagram, State Diagram을 통해 본 시스템의 구조를 설명한다

* 1. Class Diagram



Diagram

* + 1. DBHandler
       1. Attributes

UserAccessToken: 유저에게 부여된 고유한 식별번호

PlayerID: 플레이어의 ID

GamesTotal: 총 플레이한 게임 수

GamesWon: 이긴 게임의 수

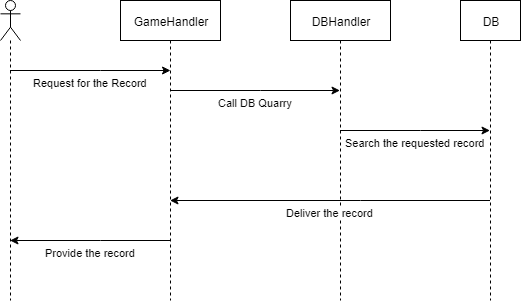
GamesLost: 진 게임의 수

SurvivalRatio: 게임 내 생존율

* + 1. UserRecord : JSON Object

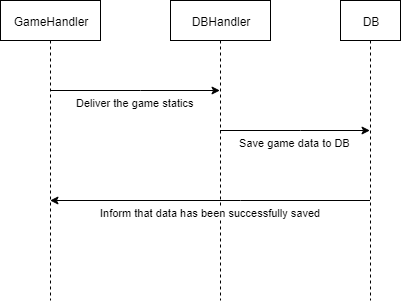
해당없음

* 1. Sequence Diagram
     1. DB Quarry Sequence : 플레이어가 자신의 전적을 요청했을 때



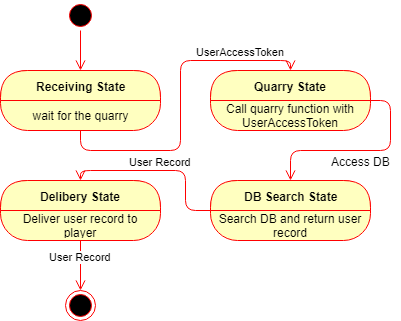
Diagram

* + 1. DB Save Sequence : 게임 종료 후 플레이어들의 전적을 갱신했을 때



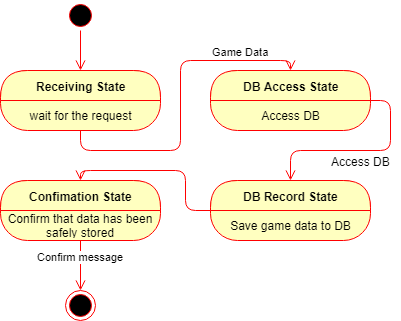
Diagram

* 1. State Diagram
     1. DB Quarry State



Diagram

* + 1. DB Save State



Diagram

1. Protocol Design
   1. Objectives
   2. JSON
   3. Protocol Description
      1. Overview
      2. .....
      3. ...
2. Database Design
   1. Objectives
   2. ER Diagram
      1. Entity
      2. Relationship
   3. Relation Schema
3. Testing Plan
   1. Objectives
   2. Testing Policy
   3. Test Case
      1. Start & End System
      2. Question & Answer System
      3. Elimination System
      4. Help & Log System
4. Development Environment
   1. Objectives
   2. Programming Language & IDE
      1. Programming Language
      2. IDE
   3. Coding Rule
   4. Version Management Tool
5. Develop Plan
   1. Objectives
6. Index
   1. Table Index
   2. Figure Index
   3. Diagram Index
7. Reference
8. wikipedia UML, 2018.05.17, <https://ko.wikipedia.org/wiki/통합\_모델링\_언어>
9. wikpedia UML, 2018.05.17, < https://en.wikipedia.org/wiki/Package\_diagram>