DAWN

Requirement Define

Version 1.0

**By:**

Dawn Group

2019-04

**Group Member:**

Zihan Xu

Yi Kuang

Chenyu Yang

Yuting Lan

Jianzhen Cao

**Document Language:**

English

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 2019-4-2 | 1.0 | Finish the 1st edition of feasibility analysis report | Zihan Xu, Yi Kuang, Chenyu Yang, Yuting Lan, Jianzhen Cao |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[1引言 1](#_Toc5438543)

[1.1编写目的 1](#_Toc5438544)

[1.2适用范围 1](#_Toc5438545)

[1.3定义 1](#_Toc5438546)

[1.4参考资料 1](#_Toc5438547)

[1.5概述 1](#_Toc5438548)

[2目前系统 1](#_Toc5438549)

[3建议的系统 2](#_Toc5438550)

[3.1概述 2](#_Toc5438551)

[3.2功能需求 2](#_Toc5438552)

[3.2.1 <功能需求一> 2](#_Toc5438553)

[3.3非功能需求 2](#_Toc5438554)

[3.3.1可用性 2](#_Toc5438555)

[3.3.2可靠性 2](#_Toc5438556)

[3.3.3性能 3](#_Toc5438557)

[3.3.4可支持性 3](#_Toc5438558)

[3.3.5设计约束 3](#_Toc5438559)

[3.3.6 接口 4](#_Toc5438560)

[3.3.7 法律、版权及其他声明 4](#_Toc5438561)

[3.3.8 适用的标准 4](#_Toc5438562)

[3.4 系统模型 5](#_Toc5438563)

[3.4.1场景 5](#_Toc5438564)

[3.4.2用例模型 5](#_Toc5438565)

[3.4.3对象模型 5](#_Toc5438566)

[3.4.4动态模型 6](#_Toc5438567)

[3.4.5 用户界面 7](#_Toc5438568)

软件需求规约

# 1引言

## 1.1编写目的

说明：编写这份软件需求规约的目的，并指出预期的读者。

## 1.2适用范围

列出此文档适用的软件应用程序、特性或其他子系统分组、与其相关的用例模型，以及受到此文档影响的任何其他产品。

## 1.3定义

列出本文件中用到的专门术语的定义和外文的首字母缩写词。可以引用项目词汇表来提供。

## 1.4参考资料

列出本文中各处引用的文档资料，包括每个文档的标题、文档编号、发表日期和出版单位并列出能够得到这些文件资料的来源。

## 1.5概述

说明本文件中其他各部分包含的内容，与本文件的内容组织方式。

# 2目前系统

描述目前的状态。如果目前软件项目是开发一个新的系统替代以前的系统，那么此章节可以用来描述老系统的功能和存在的问题。如果目前并没有老系统，那么该章节就描述在实际业务中遇到的一些问题。

# 3建议的系统

## 3.1概述

简要介绍本项目包含的所有软件需求。

## 3.2功能需求

说明：列出本项目的功能性需求。对于应用程序来说，此节应重点描述。在表述时，可以一一列举各条功能需求，如<功能需求一>

建议仔细考虑此节的组织方式，如按照软件特性、用户集、子系统等组织方式进行说明。通常需要说明特性、性能和安全性。

当利用应用程序开发工具（如需求工具、建模工具等）来获取功能性时，此节文档将引用获取相应数据的方法，并指出用来获取数据的工具的位置和名称。

### 3.2.1 <功能需求一>

对该条功能需求进行详细说明。

## 3.3非功能需求

### 3.3.1可用性

说明：列出所有可能影响本项目可用性的需求，例如：

* + 普通用户和高级用户熟练操作所需培训时间。
  + 可根据用现有系统来确定新系统的可用性需求。
  + 符合公认的可用性标准（如IBM的CUA标准和Microsoft的GUI标准）方面的需求。

在表述时，可以一一列举各条可用性需求，如<可用性需求一>，以下各节非功能性需求均可如此表述。

#### 3.3.1.1 <可用性需求一>

对该条可用性需求进行详细说明。

### 3.3.2可靠性

列出所有关于本项目系统可靠性的需求，例如：

* + 系统开放时间与操作权限。
  + 平均故障间隔时间、平均修复时间与精确度需求。
  + 最高代码错误率，通常表示为每千行代码的错误数目 (bugs/KLOC) 或每个功能点的错误数目 (bugs/function-point
  + 错误率，按照小错误、大错误和严重错误来分类。需求中必须对“严重”错误进行界定，例如：数据完全丢失或完全不能使用系统的某部分功能。

#### 3.3.2.1 <可靠性需求一>

对该条可靠性需求进行详细说明。

### 3.3.3性能

此节应概述系统的性能特征。其中需包括具体的响应时间。如果可行，按名称引用相关用例。例如：

* + 对事务的响应时间（平均、最长）
  + 吞吐量，例如每秒处理的事务数
  + 容量，例如系统可以容纳的客户或事务数
  + 降级模式（当系统以某种形式降级时可接受的运行模式）
  + 资源利用情况，如内存、磁盘、通信等

#### 3.3.3.1 <性能需求一>

对该条性能需求进行详细说明。

### 3.3.4可支持性

此节应列出将提高所构建系统的可支持性或可维护性的所有需求，其中包括编码标准、命名约定、类库、维护访问权和维护实用程序。

#### 3.3.4.1 <可支持性需求一>

对该条可支持性需求进行详细说明。

### 3.3.5设计约束

此节应列出所构建系统的所有设计约束。设计约束代表已经批准并必须遵循的设计决定。其中包括软件语言、软件流程需求、开发工具的指定用途、构架及设计约束、购买的构件、类库等

#### 3.3.5.1 <设计约束一>

对该条设计约束进行说明。

### 3.3.6 接口

此节规定应用程序必须支持的接口/界面。它应非常具体，包含协议、端口和逻辑地址等，以便于按照接口/界面需求开发并检验软件。

#### 3.3.6.1用户界面

说明软件将实现的用户界面方面的要求。

#### 3.3.6.2 硬件接口

此节指出软件所支持的所有硬件接口，其中包括逻辑结构、物理地址、预期行为等

#### 3.3.6.3 软件接口

此节说明软件系统中与其他构件之间的软件接口。这些构件可以是购入的构件、取自其他应用程序重新利用的构件，也可以是为此需求范围之外的子系统开发，但该软件应用程序必须与之交互的构件

#### 3.3.6.4通信接口

说明与其他系统或设备（如局域网、远程串行设备等）的所有通信接口

### 3.3.7 法律、版权及其他声明

此节说明软件涉及的所有必需的法律免责声明、保证、版权声明、专利声明、字标、商标或徽标符合性问题

### 3.3.8 适用的标准

此节说明了所有适用的标准以及适用于所述系统的相应标准的具体部分。例如，其中可以包括法律、质量及法规标准；业界在可用性、互操作性、国际化、操作系统相容性等方面的标准

## 3.4 System Model

In this section, we will be talking and defining the models used in the system.

### 3.4.1 Senario

该节中针对每一个发起用例的参与者，选择代表性的场景进行描述。按照如下格式进行：

<场景一>:

* 场景名字: 为每一个场景起的具体名字
* 参与者实例：场景中涉及到的具体参与者人员
* 事件流程：按照步骤列出详细的流程

### 3.4.2 Usecase Specification

1. All the participants：

|  |  |
| --- | --- |
| Participant | Interpretation for the participant |
| User | A user who have installed our game, he/she can login and join a game. |
| Player | A login user who is playing our game. |

1. All the usecases in our system

|  |  |  |
| --- | --- | --- |
| Usecase | Usecase Level | Interpretation |
| Register | User Goal |  |
| Login | Sub-function |  |
| CreateRoom | User Goal |  |
| JoinRoom | User Goal |  |
| SelectRole | Sub-function |  |
| Prepare | Sub-function |  |
| GameProcess | User Goal |  |
| Move | Sub-function |  |
| Attack | Sub-function |  |
| Use | Sub-function |  |
| EndOfGame | Sub-function |  |
| ShowResults | User Goal |  |

1. Detailed usecase specification.
2. Register

**Use Case Name**: Register

**Scope**: System Use Case

**Level**: User Goal

**Main participants**: User

**Stakeholders and users whose concerns:** User not yet registered: successfully registered account

**Prerequisites**: Users enter the registeration interface.

**Postcondition**: User registration is successful.

**Mainstream**:

1. users click the "Register" button on the page.

2.The system receives the user's request, jumps to the registration page and displays the registration form for filling in.

3. Users fill in the registration form's user name, login password, confirmation login password, mobile phone number, e-mail, and click the "Confirm Registration" button after filling in.

4. Systematic identification of the validity of form information.

5. The system prompts for successful registration and jumps to the login interface.

**Expansion process**

Registration form information is invalid

(1) In step 3, the login password filled in by the user is illegal. The system prompts the user to fill in the login password too short, too long or too simple, and asks the user to modify a legitimate password and submit it it again.

(2) In the third step, the user fills in the confirmation password and the login password are inconsistent.

(3) In step 3, the user name filled out by the user has been registered.

(a). The system prompts the user that the username has been registered, and requests the user to revise it, submit it it again and try again.

(b)The system prompts users to fill in inconsistent login passwords twice. Please check the changes and submit them again.

(4) In step 3, the number of the mobile phone that the user filled in has been registered. The system prompts the user that the mobile phone number has been registered. Please check the modification and submit it again.

(5) In step 3, the user's e-mail has been registered. The system prompts the user that the e-mail has been registered, and asks the user to check the modification and resubmit it.

**Special Needs**: None.

**Frequency:** It may happen at any time, but in general, a user will only register once, so the rate will not be too high.

1. Login

**Scope**: System Use Case

**Level**: Subfunctionl

**Main participants**: User

**Stakeholders and users whose concerns:** User who had registered: login for playing the game.

**Prerequisites**: Users enter the login interface.

**Postcondition**: the log of the user is successful.

**Mainstream**:

1. The user clicks the login button on the page.

2. The system receives the user's request, jumps to the login page, and displays the login form for filling in.

3. The user fills in the user name and login password of the login form, and then clicks the "login" button.

4. Systematic identification of the validity of form information.

5. The system prompts for successful login and jumps to the game selection page.

**Expansion process**

The login form information is invalid:

(1) In step 3, the username and password filled in by the user are illegal.

(2) In step 3, the username entered by the user does not exist. The system prompts the user that the username does not exist. Please ask the user check the changes and resubmit them.

(3) In step 3, the login password of the user is incorrect. User name or login prompted by the system.The password is incorrect. Please check the changes and resubmit them

**Special Needs**: None.

**Frequency:** It may happen at any time and with high frequency.

1. CreateRoom

**Use Case Name**: CreateRoom

**Scope**: System Use Case

**Level**: User Goal

**Main participants**: User

**Stakeholders and users whose concerns:** User who has logged in and is preparing to set up a game: successfully set up a game ,in another word ,successfully create a gameroom.

**Prerequisites**: Users enter the game boundary interface.

**Postcondition**: The game room has been set up successfully.

**Mainstream**:

1. users click the "CreateRoom" button on the boundary interface.

2.The system receives the user's request, jumps to the CreateRoom page and displays required choices for users，such how many people the room contain.

3. Users complete the required choices and press "Confirm CreateRoom" button after that.

4. Systematic identification of the validity .

5. The system prompts for successful CreateRoom and jumps to the room interface.

**Expansion process**

CreateRoom choices is illegal

(1) In step 3, the fundamental information set by the user is illegal. The system prompts the user to the number of the users the room can contain is beyond the limit and asks the user to modify a legitimate password and submit it it again.

**Special Needs**: None.

**Frequency:** It may happen at any time. It is unavoidable to create a Room when a game begins.

1. JoinRoom

**Use Case Name**: JoinRoom

**Scope**: System Use Case

**Level**: User Goal

**Main participants**: User

**Stakeholders and users whose concerns:** User who had logged in: join an existed room to play a game.

**Prerequisites**: Users login their account.

**Postcondition**: User join room successfully.

**Mainstream**:

1. Users click the "JoinRoom" button on the page.

2. The system receives the user's request, jumps to the room list page, searched the rooms that have been created and displays the list of rooms for users to choose.

3. Users choose a room, and click "join" button to join the room.

4. The system adds users to the room and jumps to the room page.

**Expansion process**

No room existed:

1. In step 2, no room has been created. The system prompts the user to create a room.

Failed to join room:

1. In step 4, an error occurs such as the internet instability when the user join the room. The System prompts the user the failure of join room, prompts the user to try again or choose another room and reload the reloads the list of rooms.

**Special Needs**: None.

**Frequency:** It may happen at any time, but in general, the expectation of times that a user join a room per game is less than 1, so the rate will not be too high.

1. SelectRole

**Use Case Name**: SelectRole

**Scope**: System Use Case

**Level**: Subfunction

**Main participants**: User

**Stakeholders and users whose concerns:** User who had joined a room: select a role that the user will use in the next game.

**Prerequisites**: Users join an existed room successfully.

**Postcondition**: The user selects a role successfully.

**Mainstream**:

1. The user clicks the box of himself or herself on the page.
2. The system pops up a role select page that is less than the room page, which contains all the roles that the user owns.
3. The user clicks a role to select a role, click “Select” button to confirm the choice.
4. The system hides the role select page and change the role information of the user on the room page.

**Expansion process**

The user doesn’t select a role:

1. In step 1, the user doesn’t select a role and then click the “Prepare” button. After the game begins, the system will assign a random role to the user that the user owns.

The role has been chosen by the other user:

1. In step 3, the role has been selected by the other user. The system prompts the user to select again.

**Special Needs**: None.

**Frequency:** It may happen at any time after it that the user joins a room and before the user prepares..

1. Prepare

**Use Case Name**: Prepare

**Scope**: System Use Case

**Level**: Subfunction

**Main participants**: User

**Stakeholders and users whose concerns:** User who has joined a room: claim that the user has prepared to start a game.

**Prerequisites**: Users join a room.

**Postcondition**: The user prepares successfully.

**Mainstream**:

1. The user clicks “Prepare” button to claim that he or she is prepared.
2. The “Prepare” button turns to “Cancel” button. Users can click it to cancel prepare state.
3. When all the users are prepared, the game start.

**Expansion process**: None.

**Special Needs**: None.

**Frequency:** It may happen at any time after it that the user joins a room and before the game start.

1. GameProcess
2. Move

**Use Case Name**: Move

**Scope**: System Use Case

**Level**: Sub function

**Main participants**: User

**Stakeholders and users whose concerns:** User not yet registered: successfully registered account

**Prerequisites**: Player presses the moving button.

**Post-condition**: The role finished moving.

**Mainstream**:

1. Players click the direction button in the game.

2. System receives the user's request.

3. System identifies the direction for moving and judge whether this move is legal.

4. System gives the permission for moving.

5. System sends the updated data for moving.

**Expansion process**

The move is identified to be illegal (moving against obstacles or moving outside the map).

1. In step 1, player pick the illegal direction for movement;

In step 3, the system finds out that multiple players are seeking request to move into the same block.

**Special Needs**: None.

**Frequency:** It may happen at any time in a game, the precise frequency depends on the frequency of game server and network status.

1. Attack
2. Use
3. EndOfGame

**Use Case Name**: EndOfGame

**Scope**: System Use Case

**Level**: Sub-function

**Main participants**: User

**Stakeholders and users whose concerns:** Users who are still alive when the game ended.

**Prerequisites**: the game is ended.

**Post-condition**: users press the ‘exit’ button.

**Mainstream**:

1. The game logic for end of game is triggered.
2. Ending information is sent to every user.
3. Users view the information and get informed of the result.
4. All users press the ‘exit’ button and get back to the enter interference.

**Expansion process**

None

**Special Needs**: None.

**Frequency:** It happens when the game ends, so the frequency is one time for a game.

1. ShowResults

**Use Case Name**: ShowResults

**Scope**: System Use Case

**Level**: User Goal

**Main participants**: User

**Stakeholders and users whose concerns:** Users in the same game that has ended.

**Prerequisites**: Users trigger the ending condition or the game time is over.

**Post-condition**: All players have click the acknowledge button.

**Mainstream**:

1. Players have triggered the ending condition or the game time has ended.
2. System send the results to players.
3. Game show the results in the interface.
4. Players click the acknowledge button.
5. Display ended, get back to the enter interface.

**Expansion process**

None.

**Special Needs**: None.

**Frequency:** It happens when a member triggers the ending condition or the game time ends.

1. 画出完整的用例图。

### 3.4.3对象模型

[分类别列出每一个类的介绍]

表 实体类定义表

|  |  |  |  |
| --- | --- | --- | --- |
| 实体类名称 | 属性 | 关联类 | 定义 |
|  |  |  |  |
|  |  |  |  |

表 边界类定义

|  |  |
| --- | --- |
| 边界类名称 | 定义 |
|  |  |
|  |  |

表 控制类定义

|  |  |
| --- | --- |
| 控制类名称 | 定义 |
|  |  |
|  |  |

[列表方式介绍完类后，画出一张类图。]

### 3.4.4动态模型

1. 系统顺序图与操作契约

(1)用例X

* 为每一个用例画出一张系统顺序图
* 为该用例中的系统消息定义操作契约

1. 顺序图

[围绕系统消息画顺序图，为其中每一个系统消息单独画一张顺序图。]

1. 状态图

[给出某些具备复杂状态的类的状态图。]

### 3.4.5 用户界面

在需求获取阶段，可以给出部分用户界面，并在分析阶段继续完善。在介绍界面时，可以通过文字描述界面之间的转换关系。