Software Requirements Specification

Version 1.0

March 14, 2019

Android Games Publishing System

Waleed Aljrf

Hala Samaan

Shaza Hashem

# Table of Contents

[Table of Contents i](#_Toc77487619)

[List of Figures ii](#_Toc77487620)

[1.0. Introduction 1](#_Toc77487621)

[1.1. Purpose 1](#_Toc77487622)

[1.2. Scope of Project 1](#_Toc77487623)

[1.3. Glossary 2](#_Toc77487624)

[1.4. References 2](#_Toc77487625)

[1.5. Overview of Document 2](#_Toc77487626)

[2.0. Overall Description 4](#_Toc77487627)

[2.1 System Environment 4](#_Toc77487628)

[2.2 Functional Requirements Specification 5](#_Toc77487629)

[2.2.1 Gamer Use Case 5](#_Toc77487630)

[Use case: Search Games 5](#_Toc77487631)

[2.2.2 Games companies and developers Use Case 6](#_Toc77487632)

[Use case: Submit games 6](#_Toc77487633)

[2.2.3 Reviewer Use Case 7](#_Toc77487634)

[Use case: Submit Review 7](#_Toc77487635)

[2.2.4 Editor Use Cases 8](#_Toc77487636)

[Use case: Update Developer 8](#_Toc77487637)

[Use case: Update Reviewer 9](#_Toc77487638)

[Use case: Update Game 9](#_Toc77487639)

[Use case: Receive Game 10](#_Toc77487640)

[Use case: Assign Reviewer 11](#_Toc77487641)

[Use case: Receive Review 11](#_Toc77487642)

[Use case: Check Status 12](#_Toc77487643)

[Use case: Send Response 12](#_Toc77487644)

[Use case: Send Copyright 13](#_Toc77487645)

[Use case: Remove Game 14](#_Toc77487646)

[Use case: Publish Game 14](#_Toc77487647)

[2.3 User Characteristics 15](#_Toc77487648)

[2.4 Non-Functional Requirements 15](#_Toc77487649)

[3.0. Requirements Specification 17](#_Toc77487650)

[3.1 External Interface Requirements 17](#_Toc77487651)

[3.2 Functional Requirements 17](#_Toc77487652)

[3.2.1 Search Game 17](#_Toc77487653)

[3.2.2 Communicate 18](#_Toc77487654)

[3.2.3 Add Developer 18](#_Toc77487655)

[3.2.4 Add Reviewer 19](#_Toc77487656)

[3.2.5 Update Gamer 19](#_Toc77487657)

[3.2.6 Update game Status 20](#_Toc77487658)

[3.2.7 Enter Communication 20](#_Toc77487659)

[3.2.8 Assign Reviewer 21](#_Toc77487660)

[3.2.9 Check Status 21](#_Toc77487661)

[3.2.10 Send Communication 22](#_Toc77487662)

[3.2.11 Publish Game 22](#_Toc77487663)

[3.2.12 Remove Game 23](#_Toc77487664)

[3.3 Detailed Non-Functional Requirements 23](#_Toc77487665)

[3.3.1 Logical Structure of the Data 23](#_Toc77487666)

[3.3.2 Security 25](#_Toc77487667)

[Index 26](#_Toc77487668)

# List of Figures

[Figure 1 - System Environment 4](#_Toc77487669)

[Figure 2 - Game Submission Process 6](#_Toc77487670)

[Figure 3 - Editor Use Cases 8](#_Toc77487671)

[Figure 4 - Logical Structure of the game Manager Data 23](#_Toc77487672)

# 1.0. Introduction

## 1.1. Purpose

The purpose of this document is to present a detailed description of android games Publishing System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do ,. This document is intended for both the stakeholders and the developers of the system.

## 1.2. Scope of Project

This software system will be a android games Publishing System. This system will be designed to maximize the editor’s productivity by providing tools to assist in automating the game review and publishing process, which would otherwise have to be performed manually. By maximizing the editor’s work efficiency and production the system will meet the editor’s needs while remaining easy to understand and use.

More specifically, this system is designed to allow an editor to manage and communicate with a group of reviewers and developers to publish games to a public website. The software will facilitate communication between developers, reviewers, and the editor via E-Mail. Preformatted reply forms are used in every stage of the games progress through the system to provide a uniform review process;. The system also contains a relational database containing a list of developers, Reviewers, and Games.

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Active Game | The document that is tracked by the system; it is a narrative that is planned to be posted to the public website. |
| Developer | Person or games designing company submitting a game to be reviewed. |
| Database | Collection of all the information monitored by this system. |
| Editor | Person who receives games, sends them for review, and makes final judgments for publications. |
| Field | A cell within a form. |
| IGDB | The existing membership database for android games |
| Gamer | Anyone visiting the site to download and play games |
| Review | A written recommendation about the appropriateness of an game for publication; may include suggestions for improvement. |
| Reviewer | A person that examines a game and has the ability to recommend approval of the game for publication or to request that changes be made in the game. |
| Software Requirements Specification | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document. |
| Stakeholder | Any person with an interest in the project who is not a developer. |
| User | Reviewer or game developer. |

## 1.4. References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

## 1.5. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

# 2.0. Overall Description

## 2.1 System Environment

Developer

Gamer

Editor

IGDB

Games website

Game Manager

Android games Publishing System

Reviewer

Figure - System Environment

The android games Publishing System has four active actors and one cooperating system.

The developer, gamer, or Reviewer accesses the games website through the Internet. Any developer or Reviewer communication with the system is through email. The Editor accesses the entire system directly.

<< The division of the android games Publishing System into two component parts, the games website and the game Manager, is an example of using domain classes to make an explanation clearer. >>

## 2.2 Functional Requirements Specification

This section outlines the use cases for each of the active actors separately. The gamer, the developer and the reviewer have only one use case apiece while the editor is main actor in this system.

### 2.2.1 Gamer Use Case

#### Use case: Search game

**Diagram:**

Gamer

Search Game

**Brief Description**

The Gamer accesses the android games Website, searches for a game and downloads it to his/her machine.

**Initial Step-By-Step Description**

Before this use case can be initiated, the gamer has already accessed the android games Website.

1. The gamer chooses to search by developer or company name, category, or keyword.
2. The system displays the choices to the gamer.
3. The gamer selects the game desired.
4. The system presents the abstract of the game to the gamer.
5. The gamer chooses to download the game.
6. The system provides the requested game.

**Xref:** Section 3.2.1, Search game

Edit or update

Review

Active Game

Submit

Publish

Figure - Game Submission Process

The *Game Submission Process* state-transition diagram summarizes the use cases listed below. An developer submits a game for consideration. The Editor enters it into the system and assigns it to and sends it to at least three reviewers. The Reviewers return their comments, which are used by the Editor to make a decision on the game. Either the game is accepted as designed ,or declined or the developer is asked to make some changes based on the reviews. If it is accepted, possibly after a revision , the Editor sends a copyright form to the developer. When that form is returned, the game is published to the games website .

### 2.2.2 Developer Use Case

May be one person or company..

#### Use case: Submit Game

Developer

Submit Game

**Diagram:**

**Brief Description**

The developer either submits an original game or resubmits an edited game.

**Initial Step-By-Step Description**

Before this use case can be initiated, the developer has already connected to the android games Website.

1. The developer chooses the *Email Editor* button.
2. The System uses the *sendto* HTML tag to bring up the user’s email system.
3. The developer fills in the Subject line and attaches the files as directed and emails them.
4. The System generates and sends an email acknowledgement.

**Xref:** Section 3.2.2, Communicate

### 2.2.3 Reviewer Use Case

#### Use case: Submit Review

**Diagram:**

Reviewer

Submit Review

**Brief Description**

The reviewer submits a review of a game.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Reviewer has already connected to android games Website.

1. The Reviewer chooses the *Email Editor* button.
2. The System uses the *sendto* HTML tag to bring up the user’s email system.
3. The Reviewer fills in the Subject line and attaches the file as directed and emails it.
4. The System generates and sends an email acknowledgement.

**Xref:** Section 3.2.2, Communicate

### 2.2.4 Editor Use Cases

The Editor has the following sets of use cases:

Update Info

Editor

Handle game

Check Status

Send Rec

Publish game

Figure - Editor Use Cases

**Update Information use cases**

#### Use case: Update developer

**Diagram:**

Editor

Update Developer

**Brief Description**

The Editor enters a new developer or updates information about a current developer.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the game Manager.

1. The Editor selects to *Add/Update developer*.
2. The system presents a choice of adding or updating.
3. The Editor chooses to add or to update.
4. If the Editor is updating a developer, the system presents a list of developers to choose from and presents a grid filling in with the information; else the system presents a blank grid.
5. The Editor fills in the information and submits the form.
6. The system verifies the information and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

#### Use case: Update Reviewer

**Diagram:**

Editor

Update Reviewer

IGDB

**Brief Description**

The Editor enters a new Reviewer or updates information about a current Reviewer.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the game Manager.

1. The Editor selects to *Add/Update Reviewer*.
2. The system presents a choice of adding or updating.
3. The Editor chooses to add or to update.
4. The system links to the IG Database.
5. If the Editor is updating a Reviewer, the system and presents a grid with the information about the Reviewer; else the system presents list of members for the editor to select a Reviewer and presents a grid for the person selected.
6. The Editor fills in the information and submits the form.
7. The system verifies the information and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.4, Add Reviewer; Section 3.2.5, Update gamer

#### Use case: Update game

**Diagram:**

Editor

Update game

**Brief Description**

The Editor enters information about an existing game.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the game Manager.

1. The Editor selects to *Update game*.
2. The system presents s list of active games.
3. The system presents the information about the chosen game.
4. The Editor updates and submits the form.
5. The system verifies the information and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.6, Update game Status

**Handle game use cases**

#### Use case: Receive game

Editor

Receive game

**Diagram:**

**Brief Description**

The Editor enters a new or revised game into the system.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the game Manager and has a file containing the game available.

1. The Editor selects to *Receive game*.
2. The system presents a choice of entering a new game or updating an existing one
3. The Editor chooses to add or to update.
4. If the Editor is updating an game, the system presents a list of games to choose from and presents a grid for filling with the information; else the system presents a blank grid.
5. The Editor fills in the information and submits the form.
6. The system verifies the information and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.7, Enter Communication

#### Use case: Assign Reviewer

This use case extends the *Update game* use case.

**Diagram:**

Editor

Assign Reviewer

IGDB

**Brief Description**

The Editor assigns one or more reviewers to an game.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the game using the *Update game* use case.

1. The Editor selects to *Assign Reviewer*.
2. The system presents a list of Reviewers with their status (see data description is section 3.3 below).
3. The Editor selects a Reviewer.
4. The system verifies that the person is still an active member using the IG Database.
5. The Editor repeats steps 3 and 4 until sufficient reviewers are assigned.
6. The system emails the Reviewers, attaching the game and requesting that they do the review.
7. The system returns the Editor to the *Update game* use case.

**Xref:** Section 3.2.8, Assign Reviewer

#### Use case: Receive Review

Editor

Receive Review

This use case extends the *Update game* use case.

**Diagram:**

**Brief Description**

The Editor enters a review into the system.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the game using the *Update game* use case.

1. The Editor selects to *Receive Review*.
2. The system presents a grid for filling with the information.
3. The Editor fills in the information and submits the form.
4. The system verifies the information and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.7, Enter Communication

**Check Status use case:**

#### Use case: Check Status

**Diagram:**

Editor

Check Status

**Brief Description**

The Editor checks the status of all active games.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the game Manager.

1. The Editor selects to *Check Status*.
2. The system returns a scrollable list of all active games with their status (see data description in section 3.3 below).
3. The system returns the Editor to the game Manager main page.

**Xref:** Section 3.2.9, Check Status

**Send Recommendation use cases:**

#### Use case: Send Response

This use case extends the *Update game* use case.

**Diagram:**

Editor

Send Response

**Brief Description**

The Editor sends a response to developer

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update game* use case.

1. The Editor selects to *Send Response*.
2. The system calls the email system and puts the developer email address in the Recipient line and the name of the game on the subject line.
3. The Editor fills out the email text and sends the message.
4. The system returns the Editor to the game Manager main page.

**Xref:** Section 3.210, Send Communication

#### Use case: Send Copyright

This use case extends the *Update game* use case.

**Diagram:**

**Brief Description**

Editor

Send Copyright

The Editor sends a copyright form to a developer .

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Send Copyright*.
2. The system calls the email system and puts the developers email address in the Recipient line, the name of the game on the subject line, and attaches the copyright form.
3. The Editor fills out the email text and sends the message.
4. The system returns the Editor to the game Manager main page.

**Xref:** Section 3.2.10, Send Communication

#### Use case: Remove game

This use case extends the *Update game* use case.

Editor

Remove game

**Diagram:**

**Brief Description**

The Editor removes a game from the active category.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the game using the *Update game* use case.

1. The Editor selects to remove a game from the active database.
2. The system provides a list of games with the status of each.
3. The Editor selects an game for removal.
4. The system removes the game from the active game database and returns the Editor to the game Manager main page.

**Xref:** Section 3.2.12, Remove game

**Publish game use case:**

#### Use case: Publish game

This use case extends the *Update game* use case.

Editor

Publish game

**Diagram:**

**Brief Description**

The Editor transfers an accepted game to the android games website.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the game using the *Update game* use case.

1. The Editor selects to *Publish game*.
2. The system transfers the article to the games website and updates the search information there.
3. The system removes the game from the active game database and returns the Editor to the game Manager home page.

**Xref:** Section 3.2.11, Publish game

<< Since three of the actors only have one use case each, the summary diagram only involves the Editor. Adapt the rules to the needs of the document rather than adapt the document to fit the rules. >>

## 2.3 User Characteristics

The gamer is expected to be Internet literate and be able to use a search engine. The main screen of the android games Website will have the search function and a link to “developer/Reviewer Information.”

The developer and Reviewer are expected to be Internet literate and to be able to use email with attachments.

The Editor is expected to be Windows literate and to be able to use button, pull-down menus, and similar tools.

The detailed look of these pages is discussed in section 3.2 below.

## 2.4 Non-Functional Requirements

The android games website will be on a server with high speed Internet capability. . The software developed here assumes the use of a tool such as Tomcat for connection between the Web pages and the database. The speed of the gamers connection will depend on the hardware used rather than characteristics of this system.

The game Manager will run on the editor’s PC and will contain an Access database. Access is already installed on this computer and is a Windows operating system.

# 3.0. Requirements Specification

## 3.1 External Interface Requirements

The only link to an external system is the link to the IG Database to verify the membership of a Reviewer. The Editor believes that a society member is much more likely to be an effective reviewer and has imposed a membership requirement for a Reviewer. The IG Database fields of interest to the games Publishing Systems are member’s name, membership (ID) number, and email address ..

The *Assign Reviewer* use case sends the Reviewer ID to the IG Database and a Boolean is returned denoting membership status. The *Update Reviewer* use case requests a list of member names, membership numbers and (optional) email addresses when adding a new Reviewer. It returns a Boolean for membership status when updating a Reviewer.

## 3.2 Functional Requirements

The Logical Structure of the Data is contained in Section 3.3.1.

### 3.2.1 Search game

|  |  |
| --- | --- |
| **Use Case Name** | Search Article |
| **XRef** | Section 2.2.1, Search game  SDD, Section 7.1 |
| **Trigger** | The gamer assesses the android games Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The gamer chooses how to search the Web site. The choices are by games developer, by Category, and by Keyword. 2. If the search is by developer, the system creates and presents an alphabetical list of all developers in the database. 3. The gamer selects an developer. 4. The system creates and presents a list of all games by that developer in the database. 5. The gamer selects a game. 6. The system displays the Abstract for the game. 7. The gamer selects to download the game or to return to the games list or to the previous list. |
| **Alternative Paths** | In step 2, if the gamer selects to search by category, the system creates and presents a list of all categories in the database.   1. The gamer selects a category. 2. The system creates and presents a list of all games in that category in the database. Return to step 5.   In step 2, if the gamer selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The gamer enters a keyword or phrase. 2. The system searches the Abstracts for all games with that keyword or phrase and creates and presents a list of all such games in the database. Return to step 5. |
| **Postcondition** | The selected game is downloaded to the client machine. |
| **Exception Paths** | The gamer may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when game are published and not predefined in the android games database. |

### 3.2.2 Communicate

|  |  |
| --- | --- |
| **Use Case Name** | Communicate |
| **XRef** | Section 2.2.2, Submit Article; Section 2.2.3, Submit Review  SDD, Section 7.2 |
| **Trigger** | The user selects a *mailto* link. |
| **Precondition** | The user is on the *Communicate* page linked from the Online Journal Main Page. |
| **Basic Path** | This use case uses the *mailto* HTML tag. This invokes the client email facility. |
| **Alternative Paths** | If the user prefers to use his or her own email directly, sufficient information will be contained on the Web page to do so. |
| **Postcondition** | The message is sent. |
| **Exception Paths** | The attempt may be abandoned at any time. |
| **Other** | None |

### 3.2.3 Add Author

|  |  |
| --- | --- |
| **Use Case Name** | Add Author |
| **XRef** | Section 2.2.4, Update Author  SDD, Section 7.3 |
| **Trigger** | The Editor selects to add a new author to the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents a blank grid to enter the author information. 2. The Editor enters the information and submits the form. 3. The system checks that the name and email address fields are not blank and updates the database. |
| **Alternative Paths** | If in step 2, either field is blank, the Editor is instructed to add an entry. No validation for correctness is made. |
| **Postcondition** | The Author has been added to the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The author information includes the name mailing address and email address. |

### 3.2.4 Add Reviewer

|  |  |
| --- | --- |
| **Use Case Name** | Add Reviewer |
| **XRef** | Section 2.2.4, Update Reviewer  SDD, Section 7.4 |
| **Trigger** | The Editor selects to add a new reviewer to the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system accesses the Historical Society (HS) database and presents an alphabetical list of the society members. 2. The Editor selects a person. 3. The system transfers the member information from the HS database to the Article Manager (AM) database. If there is no email address in the HS database, the editor is prompted for an entry in that field. 4. The information is entered into the AM database. |
| **Alternative Paths** | In step 3, if there is no entry for the email address in the HS database or on this grid, the Editor will be reprompted for an entry. No validation for correctness is made. |
| **Postcondition** | The Reviewer has been added to the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The Reviewer information includes name, membership number, mailing address, categories of interest, and email address. |

### 3.2.5 Update Person

|  |  |
| --- | --- |
| **Use Case Name** | Update Person |
| **XRef** | Sec 2.2.4 Update Author; Sec 2.2.4 Update Reviewer  SDD, Section 7.5 |
| **Trigger** | The Editor selects to update an author or reviewer and the person is already in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The Editor selects Author or Reviewer. 2. The system creates and presents an alphabetical list of people in the category. 3. The Editor selects a person to update. 4. The system presents the database information in grid form for modification. 5. The Editor updates the information and submits the form. 6. The system checks that required fields are not blank. |
| **Alternative Paths** | In step 5, if any required field is blank, the Editor is instructed to add an entry. No validation for correctness is made. |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the person is not already in the database, the use case is abandoned. In addition, the Editor may abandon the operation at any time. |
| **Other** | This use case is not used when one of the other use cases is more appropriate, such as to add an article or a reviewer for an article. |

### 3.2.6 Update Article Status

|  |  |
| --- | --- |
| **Use Case Name** | Update Article Status |
| **XRef** | Section 2.2.4, Update Article  SDD, Section 7.6 |
| **Trigger** | The Editor selects to update the status of an article in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen and the article is already in the database. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all active articles. 2. The Editor selects the article to update. 3. The system presents the information about the article in grid format. 4. The Editor updates the information and resubmits the form. |
| **Alternative Paths** | In step 4, the use case *Enter Communication* may be invoked. |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the article is not already in the database, the use case is abandoned. In addition, the Editor may abandon the operation at any time. |
| **Other** | This use case can be used to add categories for an article, to correct typographical errors, or to remove a reviewer who has missed a deadline for returning a review. It may also be used to allow access to the named use case to enter an updated article or a review for an article. |

### 3.2.7 Enter Communication

|  |  |
| --- | --- |
| **Use Case Name** | Enter Communication |
| **XRef** | Section 2.2.4, Receive Article; Section 2.2.4, Receive Review  SDD, Section 7.7 |
| **Trigger** | The Editor selects to add a document to the system. |
| **Precondition** | The Editor has accessed the Article Manager main screen and has the file of the item to be entered available. |
| **Basic Path** | 1. The Editor selects the article using the *3.2.6, Update Article Status* use case. 2. The Editor attaches the file to the grid presented and updates the respective information about the article. 3. When the Editor updates the article status to indicate that a review is returned, the respective entry in the Reviewer table is updated. |
| **Alternative Paths** | None |
| **Postcondition** | The article entry is updated in the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | This use case extends *3.2.6, Update Article Status* |

### 3.2.8 Assign Reviewer

|  |  |
| --- | --- |
| **Use Case Name** | Assign Reviewer |
| **XRef** | Section 2.2.4, Assign Reviewer  SDD, Section 7.8 |
| **Trigger** | The Editor selects to assign a reviewer to an article. |
| **Precondition** | The Editor has accessed the Article Manager main screen and the article is already in the database. . |
| **Basic Path** | 1. The Editor selects the article using the *3.2.6, Update Article Status* use case. 2. The system presents an alphabetical list of reviewers with their information. 3. The Editor selects a reviewer for the article. 4. The system updates the article database entry and emails the reviewer with the standard message and attaches the text of the article without author information. 5. The Editor has the option of repeating this use case from step 2. |
| **Alternative Paths** | None. |
| **Postcondition** | At least one reviewer has been added to the article information and the appropriate communication has been sent. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | This use case extends *3.2.6, Update Article Status.* The Editor, prior to implementation of this use case, will provide the message text. |

### 3.2.9 Check Status

|  |  |
| --- | --- |
| **Use Case Name** | Check Status |
| **XRef** | Section 2.2.4, Check Status  SDD, Section 7.9 |
| **Trigger** | The Editor has selected to check status of all active articles. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system creates and presents a list of all active articles organized by their status. 2. The Editor may request to see the full information about an article. |
| **Alternative Paths** | None. |
| **Postcondition** | The requested information has been displayed. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The editor may provide an enhanced list of status later. At present, the following categories must be provided:   1. Received but no further action taken 2. Reviewers have been assigned but not all reviews are returned (include dates that reviewers were assigned and order by this criterion). 3. Reviews returned but no further action taken. 4. Recommendations for revision sent to Author but no response as of yet. 5. Author has revised article but no action has been taken. 6. Article has been accepted and copyright form has been sent. 7. Copyright form has been returned but article is not yet published.   A published article is automatically removed from the active article list. |

### 3.2.10 Send Communication

|  |  |
| --- | --- |
| **Use Case Name** | Send Communication |
| **XRef** | Section 2.2.4, Send Response; Section 2.2.4, Send Copyright  SDD, Section 7.10 |
| **Trigger** | The editor selects to send a communication to an author. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents an alphabetical list of authors. 2. The Editor selects an author. 3. The system invokes the Editor’s email system entering the author’s email address into the *To:* entry. 4. The Editor uses the email facility. |
| **Alternative Paths** | None. |
| **Postcondition** | The communication has been sent. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The standard copyright form will be available in the Editor’s directory for attaching to the email message, if desired. |

### 3.2.11 Publish Article

|  |  |
| --- | --- |
| **Use Case Name** | Publish Article |
| **XRef** | Section 2.2.4, Publish Article  SDD, Section 7.11 |
| **Trigger** | The Editor selects to transfer an approved article to the Online Journal. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of the active articles that are flagged as having their copyright form returned. 2. The Editor selects an article to publish. 3. The system accesses the Online Database and transfers the article and its accompanying information to the Online Journal database. 4. The article is removed from the active article database. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is properly transferred. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article information should be archived somewhere. |

### 3.2.12 Remove Article

|  |  |
| --- | --- |
| **Use Case Name** | Remove Article |
| **XRef** | Section 2.2.4, Remove Article  SDD, Section 7.12 |
| **Trigger** | The Editor selects to remove an article from the active article database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system provides an alphabetized list of all active articles. 2. The editor selects an article. 3. The system displays the information about the article and requires that the Editor confirm the deletion. 4. The Editor confirms the deletion. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is removed from the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article and its information information should be archived somewhere. |

## 3.3 Detailed Non-Functional Requirements

### 3.3.1 Logical Structure of the Data

The logical structure of the data to be stored in the internal Article Manager database is given below.

Review

Reviewer

Article

Author

writes

sent to

writes

has

Figure - Logical Structure of the Article Manager Data

The data descriptions of each of these data entities is as follows:

**Author Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of principle author |  |
| Email Address | Text | Internet address |  |
| Article | Pointer | Article entity | May be several |

**Reviewer Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of principle author |  |
| ID | Integer | ID number of Historical Society member | Used as key in Historical Society Database |
| Email Address | Text | Internet address |  |
| Article | Pointer | Article entity of | May be several |
| Num Review | Integer | Review entity | Number of not returned reviews |
| History | Text | Comments on past performance |  |
| Specialty | Category | Area of expertise | May be several |

**Review Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Article | Pointer | Article entity |  |
| Reviewer | Pointer | Reviewer entity | Single reviewer |
| Date Sent | Date | Date sent to reviewer |  |
| Returned | Date | Date returned; null if not returned |  |
| Contents | Text | Text of review |  |

**Article Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of Article |  |
| Author | Pointer | Author entity | Name of principle author |
| Other Authors | Text | Other authors is any; else null | Not a pointer to an Author entity |
| Reviewer | Pointer | Reviewer entity | Will be several |
| Review | Pointer | Review entity | Set up when reviewer is set up |
| Contents | Text | Body of article | Contains Abstract as first paragraph. |
| Category | Text | Area of content | May be several |
| Accepted | Boolean | Article has been accepted for publication | Needs Copyright form returned |
| Copyright | Boolean | Copyright form has been returned | Not relevant unless Accepted is True. |
| Published | Boolean | Sent to Online Journal | Not relevant unless Accepted is True. Article is no longer active and does not appear in status checks. |

The Logical Structure of the data to be stored in the Online Journal database on the server is as follows:

**Published Article Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of Article |  |
| Author | Text | Name of one Author | May be several |
| Abstract | Text | Abstract of article | Used for keyword search |
| Content | Text | Body of article |  |
| Category | Text | Area of content | May be several |

### 3.3.2 Security

The server on which the Online Journal resides will have its own security to prevent unauthorized *write*/*delete* access. There is no restriction on *read* access. The use of email by an Author or Reviewer is on the client systems and thus is external to the system.

The PC on which the Article Manager resides will have its own security. Only the Editor will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the editor with *write* access to the Online Journal to publish an article.