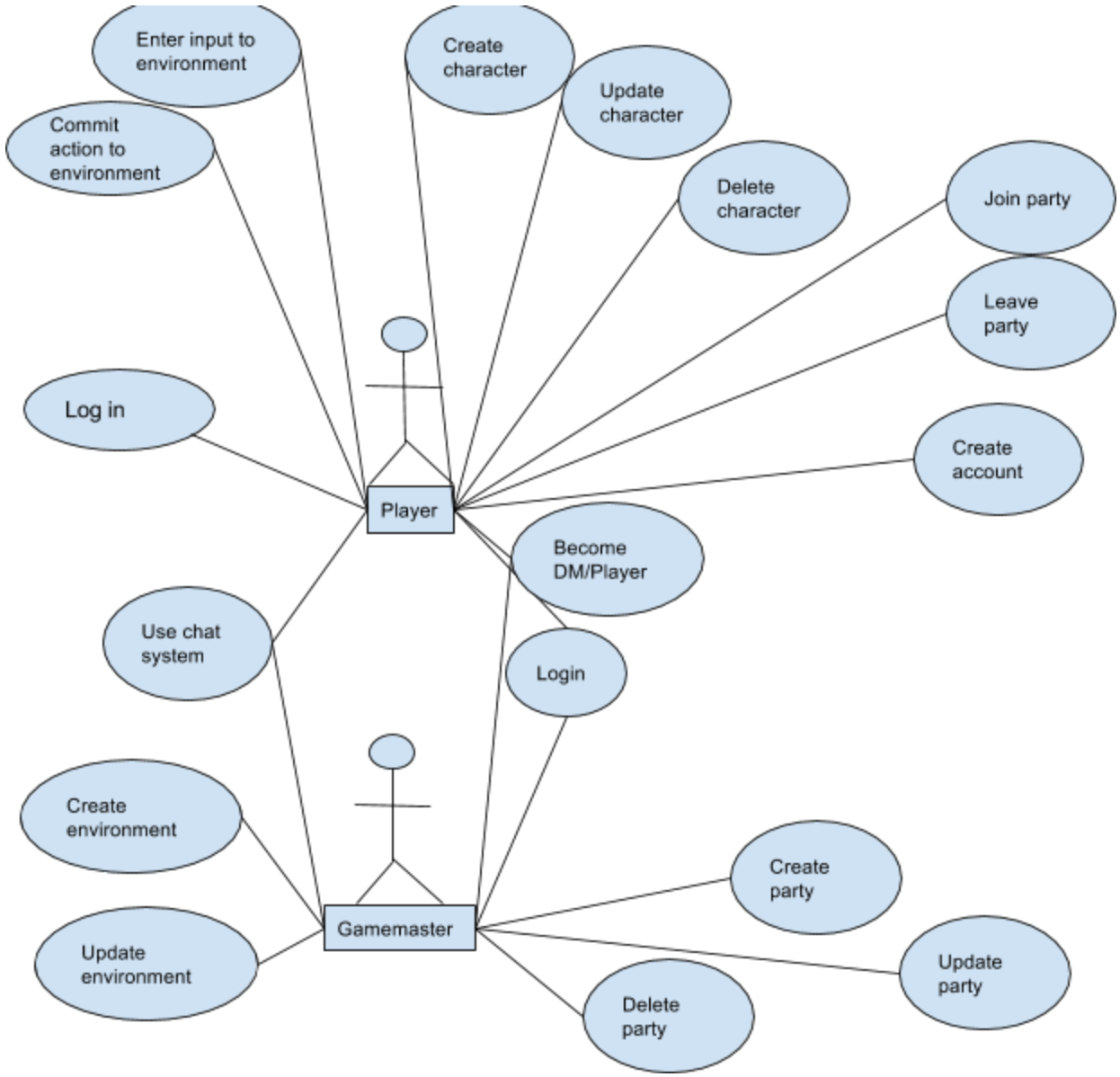
Group 13: Charles Chatwin, Michael Hartzell, Kristoffer Schindele, Logan Brewer
Utility-DC
https://github.com/frostborn4/CS386-Utility-DC.git
Deliverable 2.2
CS386 Software Engineering
Spring Term, 2/18/2017
Professor Marco Gerosa

Group 13
Use-case Specification: D&D Utility

Use Case Diagram



User Case Specifications

Completed by: Charles Chatwin

1 Brief Description

All users would be able to interact in an online environment

2 Actor Brief Descriptions

2.1 User

3 Preconditions

There is a computer available to use the product.
There is internet access for using the databases.

4 Basic Flow of Events

1. When User tries to use the program
2. They are connected to the database found online
3. The program access the database
4. The user gains all of the current information
5. The user can use or update/edit the information
6. The use case ends.

5 Alternative Flows

5.1 If No Internet

If in step 1 of the basic flow the user fails to connect, then

1. The program waits for the user to connect to the internet
2. The use case resumes at step 2

6 Subflows

6.1 If playing offline

1. User can choose to not connect, and play offline
2. Program uses only the information it last saved when online
3. User is able to use the program but unable to connect online
4. User can do anything but update or retrieve information or play with other online users

7 Key Scenarios

7.1 User disconnects

1. All other players are notified if the disconnection
2. Program skips that player's turn if the DM chooses
3. Program tries to reconnect
4. User is reconnected and everything resumes.

8 Post-conditions

8.1 User exits the program

1. User program is closed and user logged out from program

9 Special Requirements

User must have a computer or other similar device
User must have internet connection
User device must be compatible with program

Completed by: Logan Brewer

1 Brief Description

This use case describes how the user updates a character in the game.

2 Actor Brief Descriptions

2.1 Player

3 Preconditions

The player must have a character that already exists in the world.

4 Basic Flow of Events

1. The use case begins when the player tries to update his characters information.
2. The user performs an action that results in a change from the previous turn.
3. The characters information is updated to reflect the change.
4. The use case ends.

5 Alternative Flows

5.1 The player doesn't have a character

If in step 2 of the basic flow the player doesn't have a character, then

1. The user is prompted they don't have a character.
2. The user is prompted to make a new character or pass their turn.
3. The use case resumes at step 2.

5.2 The player can't make that move

If in step 2 of the basic flow the player tries to make an invalid move, then

1. The user is prompted they cannot make that action.
2. The user is prompted to perform a different action.
3. The use case resumes at step 2.

8 Post-conditions

8.1 Successful update of character information

The user made a valid move and the characters information was updated.

8.2 Failure to update character information

The character made an invalid move and either will choose a valid move or pass turn.

9 Special Requirements

The player doesn't move farther than his movement limit.
The player doesn't perform an action his character can't perform.

Completed by: Kristoffer Schindele

1 Brief Description

User creates a new character object of subtype NPC

2 Actor Brief Descriptions

2.1 Program User - a user who has already established a login

3 Preconditions

User must have established a login
User must be in GameMaster mode to create character objects

4 Basic Flow of Events

1. User logs in and switches their active mode to GameMaster.
2. User selects the 'new game object' option.
3. Then selects the 'character object' option.
4. Since all character objects owned by Gamemasters are NPCs (non-player characters) or Creature, the User selects NPC.
5. For each part of the NPC, the user is given the option of randomly generating, manually entering, or choosing from a preset list of options.
6. The User hits the save button and the newly created NPC is saved as character object under their account.

5 Alternative Flows

5.1 The User is not in GameMaster Mode

If in step 1 of the basic flow the User is not in GameMaster Mode, then
1. The only available options for character objects are Player Character and Creature.
2. The User switches to GameMaster mode and flow resumes at step 2.

6 Subflows

6.1 User Creates NPC Randomly

1. The User selects the ‘random’ option for each step of the NPC creation process.
2. The User hits the save button when completed
3. <subflow 1, step n>

6.2 User Creates NPC Manually

1. The User selects the ‘manual’ option for each step of the NPC creation process.
2. The User enters their own values for each part of the NPC creation process
3. The User hits the save button when completed

6.3 User Creates NPC by Selection

1. The User selects the ‘selection’ option for each step of the NPC creation process.
2. The User chooses from a preset list of values for each part of the NPC creation process. NOTE: not all parts of the NPC creation process can be completed via selection (example: names must either be random or manual)
3. The User hits the save button when completed

7 Key Scenarios

7.1 User quits midway through creation process

1. The program either crashes or the User otherwise exits while it is running
2. Each step of character creation updates the local copy of that character object. Information entered in the current step will not be saved. Alternatively, the User can manually save at that point and return at a later time.
3. The User opens the program again and finishes creating the NPC

8 Post-conditions

8.1 NPC is saved to local storage

The User can now access the NPC on that device, or upload it to remote storage at a later time. NPC data is saved into a text file.

8.1 NPC is saved to remote storage

A text file representing the NPC's information is saved to the web database. Other Users, including the creator can access it.

9 Special Requirements

Users can only be in the creation process of one NPC at a time, however, multiple finished and/or unfinished NPCs can be saved to storage.

Completed by: Michael Hartzell

1 Brief Description

A player can log in to their account

2 Actor Brief Descriptions

2.1 Player- somebody who wants to log into the game

3 Preconditions

A player must have an account to log in

4 Basic Flow of Events

1. The use case begins when player enters their user name

2. The player then enters their password
3. The username and password are checked against the registered users
4. The player is able to enter the game

5 Alternative Flows

5.1 Login failed

If in step 3 of the basic flow the username or password do not match, then

1. An error message is displayed
2. The use case resumes at step 1

6 Subflows

6.1 Check Password

1. For the found username get the encoded password
2. Decode the password
3. Compare the password to the typed password
4. Return the result

7 Key Scenarios

7.1 A player fails too many login attempts

1. A player has input the incorrect username and password more than 5 times
2. The player is not allowed to continue the login process
3. An e-mail is sent to the address for the given account to confirm the players identity
4. The player confirms identity

5. The player is allowed to continue logging in

8 Post-conditions

8.1 Play the game

After a player has logged into their account they have access to all the normal features. They have access to these things as long as they remain logged in.

9 Special Requirements

After a player has logged in they cannot log in to another account while that account is open. A player has to be logged in to access any other features besides account creation.

Group participation:

Charles Chatwin	Created the document, started collaboration with other members, listed the instructions, obtained the outline, filled out own use case description. Added to use case diagram.
Michael Hartzell	Created and filled out a use case diagram
Kristoffer Schindele	Created and filled out a use case diagram
Logan Brewer	Created use case diagram, filled out own use case description