Dungeons & Dragons Character Manager

**Author Name: Jamison Waters**

**Professor:** Dr. Chris Kreider

**Executive Summary:** I intend to make a character creator and manager for use with the 5th edition of the tabletop RPG Dungeons and Dragons. This desktop application will be able to manage character creation, leveling up, tracking counts of finite-usage abilities, and calculating the results of various in-game effects. This should ease the player’s burden of wrote memorization or needed 3 pages from the Player’s Handbook open at once.

Table of Contents

Introduction 4

Project Background 4

Project Description 4

Proposed Solution 4

Development Approach 4

High Level Plan 4

Project Schedule 4

Work Breakdown Structure 5

Project Calendar 5

Intermediate Milestones 5

Intermediate Milestone #1 5

Intermediate Milestone #2 5

Weekly Updates 5

Weekly Update #1 5

Weekly Update #2 5

Weekly Update #3 5

Weekly Update #4 6

Weekly Update #5 6

Weekly Update #6 6

Weekly Update #7 6

Weekly Update #8 6

Weekly Update #9 6

Gantt Chart 6

Project Deliverables 6

Conclusion 7

Appendices *<optional>* 7

Appendix A - <some resource description here> 7

# Introduction

Dungeons and Dragons is a complicated game. Every character eventually has dozens of options in a given turn, which can often lead to people forgetting their own abilities. I intend to create a program to aid people in the flow of play.

## Project Background

This project is motivated by personal experience; I want something to exist, so why not actually make it happen? I’ve had several experiences playing the game where I had forgotten I had a very useful ability, or lost track of how many spells I had left. Unfortunately, trying to manage the interplay between all the mechanics seemed to be outside my capabilities as an individual. However, over the course of several semesters, I’ve slowly accumulated understanding of various patterns and techniques that have turned this idea from an impossible monolith into a potentially manageable project.

## Project Description

I want this software product to be capable of character creation, leveling up, multiclassing, feats, combat, spell usage tracking, polymorphing, damage modifiers, and a litany of smaller things. While I will likely release this as open source, I will be developing this as a Linux desktop application, so I can’t imagine very many people actually using it any time soon.

# Proposed Solution

## Development Approach

My approach will most closely resemble the waterfall model. Dungeons and Dragons is an especially dynamic game, and a failure to accommodate for its flexibility in my own design will be especially costly, potentially having to rebuild large portions of the code base to accommodate. Additionally, everything I intend to implement is codified in my reference book, so the project requirements are subject to very little change. Along the way, I will likely need to build small prototypes for isolated systems, either to experiment with libraries or ensure a design operates as intended, but the larger structure will be built wholly after this phase is done.

## High Level Plan

At this moment, I intend to write this project in C, and I hope to use Glib to give C the flexibility needed to handle operations based on parsed data. I expect to use their dynamic arrays and strings for convenience, as well as hash maps to avoid hard-coding solutions for unusual circumstances. I will use GTK+ for the GUI frontend. For simplicity, I will likely use the XML import feature for the GUI itself, which I will design in Glade. I will of course be making said GUI. Under the hood, I will need to make the structs for the character, spells (separating race-spells and class-spells), spell lists, and abilities (separating passive from active). I will also need to make the data files my program will use as reference when bestowing a character with abilities.

# Project Schedule

## Work Breakdown Structure

The first thing to be completed is the ground-level research on how each class is defined, which I can then use to create data structures for the various character elements. After that, I will be making Lastly, I will need to make a backend to handle the data, and link it to the GUI.

## Project Calendar

### Intermediate Milestones

#### Intermediate Milestone #1

* Lists of functional requirements used to create data structures
* A UML diagram showing character data, classes, races, items, creatures, and spells, showing technical solutions to problems with agnosticism
* Flowchart of planned GUI navigation paths
* GUI mockups showing planned landing screen, character, action menu, spell lists, level-up screens

#### Intermediate Milestone #2

* GUI prototype, formatted in XML and presented in Glade
* A simple interpreted programming language used by the app to perform the functionality of various abilities.
* All data files intended for final delivery
* Command line programs that show successful parsing of data files

### Weekly Updates

#### Weekly Update #1

* Itemize requirements to implement barbarians, bards, clerics, druids, fighters, and monks (30 minutes each)
* Begin planning ways to store class role information. (2 hours)
* Gain better familiarity with GUI toolchain (3 hours)
* Create data structure for in-game items (2 hours)

#### Weekly Update #2

* Itemize requirements to implement paladins, rangers, rogues, sorcerers, warlocks, and wizards (30 minutes each)
* Incorporate new classes into class plan. This should result in a comprehensive list of every trait any given class role could ever have (2.5 hours)
* Gain better familiarity with GLib (3 hours)
* Review provided options for storing data, settle on one solution (5 hours)
* Review details of spellcasting (1 hour)

#### Weekly Update #3

* Use plans for class, race, and background to create data structures (4 hours)
* Itemize requirements of simple creatures (1 hour)
* Formulate a way to format all stored data using selected method (5 hours)
* Complete said data files for races, backgrounds (3 hours)

#### Weekly Update #4

* Itemize requirements for spells and spell lists (2 hours)
* Create data structure for spells, spell lists, simple creatures (3 hours)
* Create data files for classes (4 hours)
* Review details of adventuring and combat (2.5 hours)

#### Weekly Update #5

* Layout GUI for start page, character creation pages, character screen, turn screen, spell lists, attack screen, and damage screen (6 hours)
* Select specific spells, items, feats, creatures to implement (2 hours)
* Create interpreted language to specify requirements of abilities (4 hours)

#### Weekly Update #6

* Complete data files for classes (3 hours)
* Review and implement any minor GUI components (3 hours)
* Write GUI backend (5 hours)
* Make data files for selected spells, items, feats, creatures (1.5 hours)
* Write code to serialize character data into a data file for storage (2 hours)

#### Weekly Update #7

* Write code to read and interpret contents of data files (8 hours)
* Write code to modify current character data from parsed data files (4 hours)

#### Weekly Update #8

* Test/debug GUI (2 hours)
* Give sibling chance to use project and provide feedback (2 hours)
* Incorporating sibling feedback into project (3 hours)
* Create readme (3 hours)

#### Weekly Update #9

* Tie up any loose ends (Unknown)
* Create four characters as mock party for demonstration (2 hours)
* Plan combat encounter as final presentation (2 hours)

# Project Deliverables

Documentation

* PDF Flowchart for the theory of operation
* PDF UML diagram
* UI feedback notes

Technical Deliverables

* Link to GIT repo with completed code, GUI XML, and data files for project
* Instructions/readme document explaining how the instructor can install/test the system that was demonstrated, use basic features of the program, create custom data files to expand the database

Final Presentation Deliverables

* Four character files meant to demonstrate the diversity of features provided

# Conclusion

At the end of everything, I expect to have a functioning framework that can assist Dungeons and Dragons players with stat strategy and data management. Once it’s finished, I hope that new players will find this tool, and find it useful for managing their needs, (maybe after a Windows port.) I think the 9-week timescale is about right for this project. At the very least, it seems like both an overestimation and an underestimation. My biggest concern is, as the most complex stage of planning hasn’t been completed yet, I’ll encounter a feature I’m unprepared to accommodate. If C proves too pedantic to accomplish my goals, I may have to switch to a scripted language. I’m unsure at this point if I should store data in an SQL database, XML, or some custom binary format.

# Appendices

## Appendix A – GTK+

https://developer.gnome.org/gtk4/4.0/

## Appendix B - GLib

https://developer.gnome.org/glib/2.66/

## Appendix C – XML (maybe)

https://www.msweet.org/mxml/mxml.html

## Appendix D – SQLite (maybe)

https://sqlite.org/capi3ref.html