Final Report on the Bestiary

Martha Louise Hamlin
Northeastern University
hamlin.mar@husky.neu.edu

ABSTRACT
In this paper I discuss my semester project for Professor Timothy Bickmore's Fall 2013 Human Computer Interaction course (is4300). First I will cover the problem my software aimed to solve, including the intended users and tasks. Then I will dive into the design I came up with, examining each of the main screens and some of the decisions made along the way. Then I will move into the behind the scenes implementation, including discussion of how the design decisions shaped the data and vice versa. Then I will move on to the three main evaluations methods used throughout the development process. I will discuss the results of each evaluation and the changes that were implemented as a result, including some changes which weren't completed. Finally I will reflect on the process as a whole and the decisions involved there.

INTRODUCTION
The system I built is designed to work with Dungeon World, tabletop roleplaying game which was originally funded by a Kickstarter campaign and exists under a creative commons license. For more information regarding Dungeon World go to http://www.dungeon-world.com/.

Problem
The problem my software aims to solve is that Game Masters (GMs) of tabletop roleplaying games generally have a large number of monsters to keep track of. GMs who run games frequently could easily end up with well over 100 monsters to keep track of. Most GMs will keep these monsters on index cards, in binders, or in text files. However without good organization finding a monster will inevitably involve flipping through the whole pile, adding an unreasonable delay during play time. Attempts at organization are little better as they can quickly become cumbersome and time consuming to establish and maintain, increasing GM workload outside of the game.

Another problem I am to fix is monster creation. Dungeon World provides a monster creation survey, however using it by hand is cumbersome as each choice causes up to three changes to the existing monster framework. As the user works their way through the survey they have to continually alter the monster they are creating. This results in a high potential for error and an astonishingly time consuming process.

Users
The primary users of my software will be Dungeon World GMs. Due to the difficulty of GMing most GMs are at least in their teens. Also GMs must have a solid understanding of the system they are using, so any GM will have read through the playbook and/or played in a large number of games. Due to Dungeon World being primarily available online most GMs have at least rudimentary computer literacy. Due to the crossover between computer games and tabletop roleplaying games and general “geek” culture, most users will have a high degree of tech literacy.

The secondary users would be the GM’s players, as they will interact with the monsters created in the Bestiary. Their play sessions will be better if the GM devotes less time to finding, creating, and organizing monsters and more on planning and playing.

The tertiary users would be friends of Dungeon World players who may hear stories of the adventures and as a result decide to join later. Also included would be the creators of Dungeon World who could see an uptick in players (and thus proceeds) if the Bestiary helped Dungeon World games run more smoothly and thus be more accessible and enjoyable to new players.

The facilitating stakeholder at this time is me. As much of Dungeon World is Creative Commons if I eventually got my system to a point where I was happy with it and decided to release it under a Creative Common's license myself then the number of facilitating stakeholders could increase.

Tasks
The software was designed to fully support three compound tasks, however these tasks can be broken up into five subtasks. These task are:

- Create a monster using the monster creation survey.
- Create a monster using it's stats (health, armor, moves, etc).
- Fetch a monster from the Codex (an online monster database).

Note I will use the term “monster” throughout this report, however the system is designed to be applicable to more then just the standard “monster.” The Bestiary can easily be used with named non-player characters and even non-hostile beings.
Add a monster to a collection (for organization).

Add a GM note to a monster (some campaign related note to self).

**DESIGN**

My software is intended to give GMs a way to easily create, store, sort and share their monsters. The program is intended to work with Dungeon world, thus the monster information that can be stored is based on Dungeon World's monster style. However it could likely be used with minimal difficulty for any other Apocalypse World based system.

**Overview**

All main pages have a uniform navigation bar at the top consisting of the program title and three buttons. One button is to go to “My Bestiary,” one to create a new monster, and one to access the Codex. These buttons were picked as they are the most common actions performed. There was originally a separate “Home” button which lead to a screen with recent and popular monsters. However in the paper prototyping stage the “Home” screen was deemed redundant and cut from later implementation.

**My Bestiary (Main View)**

When you open the screen brings the user to the “My Bestiary” page (Figure 1). This page has a scrolling list of monsters and a side bar with filtering options. Filtering isn't currently implemented but in theory pressing the filter button would cull the main flow of monster so that it only showed monsters that had the checked tags or were in the selected collections. The filtering uses check boxes to reduce user memory load and prevent errors. There is no reason to allow free input as it is a filter system not a search system so the set of possible inputs is finite, and relatively small.

**Codex**

The Codex (Figure 2) is likely the weakest section of the system but that is due to external constraints. Dungeon World has an online Codex where users can upload monsters, however there is no good searching or sorting functionality. The connection to the codex isn't currently implemented but without keeping a local copy of the Codex there isn't a way to improve the search and sort functionality.

Thus working with what is availability the Codex page of my program provides the same “Recent” and “Popular” monster scrolls as the Codex website and allows the user to download the monsters. Thus adding the monsters to their own collections where they can sort them properly.

**Monster Edit**

The monster edit screen (Figure 3) allows the user to freely change any aspect of the monster's stats, as long as they conform to Dungeon World's standards. The stats are laid out in a style similar to the way monsters are described in the Dungeon World book and the “Monster Card” described later. This way a user familiar with the layout can quickly find what they are looking for.
The various methods of input provided force the information to comply to Dungeon World's style. It is essentially impossible to fill out the form in a manner that will result in an invalid monster. This is done by providing pop up windows for selecting Monster and Attack tags, as there are fixed sets in Dungeon World. Drop down menus for the type of dice to roll and how to treat the rolls confine the dice. While spinners with set ranges keep values like health and armor to positive numbers. Typed user input is very prone to errors so this style helps reduce errors.

The main changes made to this screen were after the paper prototyping phase. Originally the moves list popped out a separate window for adding and removing moves from the list, however this separate popup was deemed unnecessary and the editing ability was brought back into the monster edit screen. The other major finding was that many users were confused by the original roll style drop down which had “/b/w.” Thus it was replaced by the clearer “Normal, Best of 2, Worst of 2.”

Later in the process after heuristic evaluation and before the user testing I decided to replace the single text box of “Special Qualities” with a list element similar to moves. This decision was made when I realized that monsters tended to have a number of separate “Special Qualities” that were just displayed as a comma separated list. (As oppose to a single special quality.)

Survey Creation
Similar to the Codex the Survey Creation was slightly restricted in that the survey text and the effects each decision had was based directly on the survey in the book. It would have been a major consistency issue if I had tried to change it at all. Despite the limitations the Survey Creation process underwent a number of major changes.
Monster Focus
Another major screen is the Monster Focus screen (Figure 6). After clicking on the “Open Monster Details” button or finishing creating a monster the user is brought to this screen which shows the Monster Card along with the collections it is in and the GM note. Everything on this screen is static and saved, to change anything the user must press one of the edit buttons. This helps prevent users accidentally changing their monsters when they just want to view them. This page also contains the “Upload to Codex” button. The primary challenges in designing this page were making it clear that the collections listed were the collections the current monster was in and making it clear that all the information on this page is saved and static. These both came down to tweaking colors, boarders, and text.

Monster Popups
The last major feature wasn't added until after the Heuristic Evaluation. One evaluator commented that there was a user memory load issue in that the GM couldn't look at more than one monster at once very easily. Thus I added a function where a user could “pop up” a monster. When a user clicks to “Pop Up Monster” it opens a non-modal dialog box which shows a small Monster Card with the monster's details and the user has the option to flip the card over and see the GM note or click a button to open the monster in the Monster Focus pane. This way the GM can have any arbitrary set of monsters open at the same time, a very useful feature for GMs.

IMPLEMENTATION
My software's backend is essentially to the point of a high functioning prototype. The monster creation and editing and GUI runtime behavior is all implemented. The only internal behavior that isn't currently implemented is filtering monsters. Any sort of external interaction is currently faked. For example the connection to the Codex is just a static dummy. Similarly the system has no save or load functionality so monsters are only remembered for as long as the program is open. Some important system elements have been implemented though.

Monster Data
All monsters are stored as an instance of the monster class. A monster object contains a significant number of fields to store it's data including Strings, ints, ArrayLists, arrays, and enums. These fields hold the Monster's name, moves, attack dice, health, tags, GM notes, the collections it's in etc. Every piece of information that makes the monster what it is. Each field is private and comes with a getter and a setter, and potentially a few other methods for modifying the values. Furthermore the monster object has a number of print methods, each of which returns a specifically formatted string. In some cases there is a one to one correspondence between the fields and the print method (for example printArmor). In other cases a print method pulls from a number of fields (for example printAttackString uses information from five different fields).

Eventually these monster objects will be the primary pieces of data being saved and loaded by the program.

Tag Arrays and Filtering
One of the pieces of information about monsters that the system needs to keep track of are tags. Dungeon World has a static set of Attack Tags and Monster Tags which can
apply to any monster. These are stored in two arrays of boolean so any true value corresponds to the monster having a specific tag.

This style of saving tags is to help with filtering when it is implemented. If given a tag filter represented by an array where trues correspond to a desired tag the system will be able to check if the monster fulfills the filter by looping over each element and doing a simple logical check. That is checking if for each tag the boolean in the filter is false or the boolean in the monster is true.

Sadly collections aren't quite as simple. This design only works so well because the set of tags is finite so I don't have to worry about tags being added and removed and the lists getting out of sync. Thus is is safe to cut it down to a series of booleans. Storing lists of collections requires ArrayLists of Strings, thus filtering will require a number of contains calls.

**Monster Lists**

Monster Lists are lists of Monsters sorted in alphabetical order. They also contain various methods based on monster objects or monster name, including checking if a monster is in the list, removing a monster from the list, removing an erased collection from the monsters within, and loading a monster to a card. My system doesn't allow monsters to have identical names and MonsterLists work to enforce this. Currently a MonsterList is used to control the monster scroll on the main page. Eventually MonsterLists will also enable filtering.

**Monster Cards**

Monster cards sit somewhere between implementation and GUI. They are custom javabean that extend JPanel and contain all the information on how to display a monster. Each one holds a handle for the monster to display and uses the monster's print statements to get the strings. It takes these strings and puts them into it's various text fields, text areas, and tooltips. It also has some ability to resize fonts in an attempt to fit information. There are two standard monster cards “Large” and “Mini” these have different default sizes and default fonts.

Monster cards are used throughout the interface in other custom swing classes like the cards with buttons that show up on the main monster scroll (Figure 1), or the little popup monster windows with a card, a couple buttons, and a GM note (Figure 7). They are also used directly in the main interface in the Monster Focus screen (Figure 6). These Monster Cards help ensure Monsters are displayed neatly and consistently throughout the interface without large chunks of copy pasted code.

---

**EVALUATION**

**Paper Prototyping**

A paper prototype was tested on three users. This exposed a number of issues including a number of missing windows and buttons. However the biggest issue brought to light was confusion over the prompt window asking if the user wanted to use the monster creation survey to create their monster. Some users avoided the survey even when they were supposed to use it. Others didn't realize that if they said they didn't want to use the survey they could still create a monster.

This issue was fixed in a two pronged manner. First the prompt window was changed to ask if the user wanted to use the Monster Survey or Free Edit and explained briefly what each one was for. This way both options were explicitly provided to the user. The second change was actually made to the tasks. The tasks were reworded to tell the user if they should use the Monster Survey or the Free Edit mode for each task. This is because the purpose wasn't to test if they knew what each creation style was, it was to test if they could find the styles and the ease of use of each creation style.

There were two other issues noted that ended up stubbornly continuing through multiple rounds of testing and multiple attempts to fix. The first being confusion over the “Collections” pane on the Monster Focus screen. This pane is meant to reflect the collections the monster is currently in. Originally it was just called “Collections,” however users thought that it was supposed to contain a comprehensive list of all collections. The name was changed to “Current Collections” to attempt to alleviate that confusion. The other issue was users consistently became confused over the prompts in the monster survey to write special qualities or moves. These prompts were then replaced with more detailed prompts like “Write special quality about it's appearance.”

**Heuristic Evaluation**

Heuristic evaluations primarily highlighted a number of bugs, glitches, and missing features. There were a couple issues where fields and windows of the survey weren't being reset to default. There originally wasn't a way to delete monsters, so that was added. There were a number of visibility problems when overly long strings were loaded into Monster Cards, so I added features to resize Monster Card text and add the text to a tooltip so even if the text won't fit it is sill visible in some manner. Window resizing was also reenabled at this time with a minimum size added to prevent extremely odd sizing.

Another user commented on the inability to see multiple monsters at once except for on the main scroll, this eventually lead to the monster popup feature.
User Testing
Five users tested a more fully implemented version then the one used for Heuristic Evaluation. At this point the backend was implemented enough that data bugs began appearing, especially surrounding tags saving incorrectly. The edit collections window had a similar issue where boxes were being unchecked by the window being refreshed or where boxes would remain checked when they weren't supposed to. A way to remove collections was also added at this time.

The main complain from users though was the fact that the cursor focus didn't flow through the interface in the way they expected. The most obvious case was when the survey was advanced the cursor didn't immediately drop to the newly revealed textfields. Also when the button was pressed to add a move the focus didn't jump back to the move field. These issues, while minor caused significant hesitation and extra clicks on the user's part. Thus the movement of the cursor and focus was hardcoded into the progression of the survey and a number of other places.

On a similar vain when presented with a dialog box with a single text field users felt they should be able to hit enter while on the text box to save and close the window. They also felt they should be able to hit enter while on the new move text box to add the move, instead of pressing the add move button. Since a large percentage of users expressed some opinion on this matter I went through and added more options for mouse-less navigation.

The confusion over the “Current Collections” panel on the Monster Focus page and the prompts for Special Qualities or Moves in the Survey returned in the user testing. I attempted to fix the issue with the “Current Collections” by renaming it to “In Collections.” I ran out of time before fixing the wording of the prompts but I believe adding examples would be the best solution. There were a number of other minor and cosmetic issues with the interface that I ran out of time to fix. However I have notes on them so if I end up continuing my work on this project I will have a place to start.

Reflection
If I did this again frankly the first thing I would try to change was the size of my interface. My interface was unreasonably large considering the time constraints and the fact that I was working on it alone. With a competent group member it might have just been a slightly over sized project. Without any assistance it finishing everything I wanted to ended up being an unrealistic expectation. Simply put bit off more then I could chew with the scope of this project.

If I were to do this again at minimum I would have cut out the Codex completely so I could focus more on the rest of my interface. As I realized the scope of my project was getting out of hand the Codex was the part that I tended to starve of time and attention, so it ended up being one of the weaker parts. It probably would have been smarter with the time constraints to leave it on the cutting room floor and focus that time and effort on polishing some other element of my project, or sleeping.

I also would have worked harder to try and get my hands on more representative users. To this day despite running eight users through, some with GMing experience and others with Dungeon World experience, I have yet to run a Dungeon World GM through my tests. This was due to issues getting access to Dungeon World GMs as all of the ones I know are over in New Zealand. In some ways running the other users through was very helpful as it got me to focus on accessibility and ease of use, but I still don't know what my actual users would think.

As to the types of prototyping to use. I found the user testing (both paper prototyping and with the program) to be most useful. Seeing what the user's tried and thought while they worked and what their impressions were afterwards was where I found most of my issues. I found Heuristic Evaluation to be the least useful as much of the feedback was off point or vague and I had less ability to grill the user or see what lead to their comments.

All in all I learned a lot through the iterative design process. It was very useful getting feedback before I had even made the interface considering the amount of time I ended up sinking into the creation of the swing GUI. The feedback throughout the design process also helped me realize how many assumptions I was making regarding my user and their knowledge. Some of the assumptions weren't necessarily problematic but it was important to realize I was making them. Further the user testing helped me notice all the times I thought about implementing something then never got back to it. In the future I would definitely try to keep better track of what features I want to implement at some later point in time.