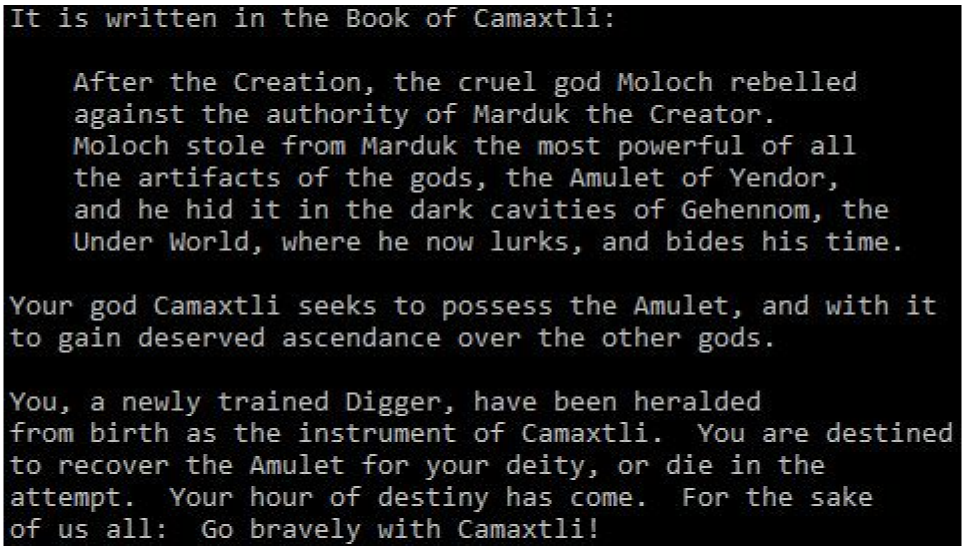
Implementation Plan for System TwitterNetHack

Assignment in the course PA1435 Objektorienterad Design

2017-04-10

Robin Westerlund, Max Nilsson, Rikard Magnom, Michael Lindroth

Author Name	Social Security Number	Thinking	Writing
Robin Westerlund	9510090195	25%	10%
Max Nilsson	9509052875	25%	0%
Rikard Magnom	9701014699	25%	90%
Michael Lindroth	9012283033	25%	0%



```
It is written in the Book of Camaxtli:

    After the Creation, the cruel god Moloch rebelled
    against the authority of Marduk the Creator.
    Moloch stole from Marduk the most powerful of all
    the artifacts of the gods, the Amulet of Yendor,
    and he hid it in the dark cavities of Gehennom, the
    Under World, where he now lurks, and bides his time.

Your god Camaxtli seeks to possess the Amulet, and with it
to gain deserved ascendance over the other gods.

You, a newly trained Digger, have been heralded
from birth as the instrument of Camaxtli.  You are destined
to recover the Amulet for your deity, or die in the
attempt.  Your hour of destiny has come.  For the sake
of us all:  Go bravely with Camaxtli!
```

System Description

NetHack is a game that involves traversing a maze-like dungeon called “Mazes of Menace”, in which the goal is to obtain the Amulet of Yendor and escape the dungeon alive. Along with the amulet you are encouraged to loot as much treasure as possible, since you want the most points possible. This will place you on the list of high scores.

The goal of the system is to modify the classic game NetHack by using Twitter to generate dungeons and implementing multiplayer support. It will also be expanded in many ways to achieve more depth in the game.

Prioritised List of Use Cases

Motivation for Priorities

1. Since the actual maze and dungeon exploration is such an integral part of the game we have prioritized the “Movement” use case very highly.
2. Seeing as we are planning to create a text based implementation the “Look Around” use case has a high priority, as the player needs to know what options are available to him.
3. Battling being the other key element of the game alongside dungeon exploring this is also prioritized quite highly.
4. Not a game if you can’t lose.
5. Roughly tied into battling, however not as essential to the gameplay.
6. Prioritized lower than battling as the feature is not useful until battling is implemented.
10. Inventory alongside the use cases 7-9 is the last key feature of the game and have a medium level of priority. The prioritization on these use cases is vague as they are very tightly connected to each other.
13. “Load Game” along with use case 12 “Save Game”, while interesting features, are non-functional features and as such are prioritized very low.

Use Cases

1. Movement (5)
2. Look Around (2)
3. Battling (5)
4. Lose (1)
5. Escaping (1)
6. Status Check (1)
7. Pick Up Item (3)
8. Manage Inventory (3)
9. Use Item (5)
10. Drop Item (1)

11. Exit Game (1)
12. Save Game (2)
13. Load Game (2)

Estimated Velocity Per Iteration

Presented below is the estimated velocity per iteration.

MAX	16
MIN	8
AVERAGE	12

Implementation Plan

Motivation for Implementation Plan

The order in which we plan to implement the system is structured the way it is because the use cases need to be implemented in a specific order. This is due to certain use cases needing other features before they can be implemented. The story points are estimated based on the smallest use cases being one point, and the others are more or the same based on their relative size.

Iterations and Use Cases

First iteration: For the first iteration we plan to implement the first four use cases (Movement, Look Around, Battling, Lose) totalling 13 story points. This should be a reasonable amount of work.

Second iteration: The second iteration will include the next six use cases (Escaping, Status Check, Pick Up Item, Manage Inventory, Use Item, Drop Item) totalling 14 story points. This is a lot of work, although it should be doable. The reasoning behind this is because Manage Inventory, Use Item and Drop item should be implemented at the same time due to being connected.

Third iteration: The third iteration will finish the system with the last three use cases (Exit, Save, Load) totalling 5 story points. This will be easily doable within the time frame if the estimation of time is correct and the previous iterations are done before this one.