RAH (RPGS At Home): Project Proposal

Samuel Xiang

0.1 Introduction

The language I would like to design would solve the ”problem” of designing turn-based RPG games on a computer. This language would be able to generate a playable game with an interactable world and combat between characters, while giving the programmer plenty of flexibility to add to the existing structure as they please. In essence, it would provide the programmer with an outlet that allows them to express complex creative ideas in simple ways.

There are a vast selection of unique turn-based RPG’s in the real world, but the genre has rules and restrictions of form that are almost impossible to break. As such, expressing the basic tenets of any turn-based RPG can be boiled down to several patterns, which mean that a formal grammar is perfectly capable of encapsulating these tenets. However, building these core ideas over and over again is redundant and time consuming, so a programming language should be a fun way to create these games with minimal labor.

0.2 Design Principles

This language should express lengthy ideas and systems of turn-based RPG games in simple English. From the technical perspective, this program should generate a playable game to the users specifications within the scope of the language. By playable, I mean a game that an outsider user interact with and complete without knowledge of the inner workings - like an actual video game. Because these ideas are expressable in simple English, I would like to create a readable program without lots of weird characters that almost anyone could access and easily intuit. Because the creation of a game involves the creation of a fictional world, the program should almost read as a story or encyclopedia of this world.

0.3 Examples

My language expects a text file of the .rpg format. Example: dotnet run ”gamename.rpg”
Example 1. To run, type ’dotnet run dnd.rpg’ in the console.

There is a room named ”Nautiloid” with description: ”The nautiloid destroys all conceptions of engineering and construction that you understand. It seems like you’re standing on a living, breathing creature. Every surface pulsates and pus oozes underfoot.” with objects: (”Playable” character named ”Tav” with stats: hp = 100, mp = 50, atk = 25, def = 15, matk = 25, mdef = 15, spd = 15 and abilities:) and connections: ”South” is ”Causeway”.

There is a room named ”Causeway” with description: ”The raised path is surrounded by a viscous pool of a mysterious, sanguine liquid. There is a path to the east and an opening above you.” with objects: (”Nonplayable” character named ”Mindflayer” with stats: hp = 150, mp = 100, atk = 25, def = 10, matk = 50, mdef = 50, spd = 20 and abilities: ability named ”Flay” has effect ”deal 100 matk dmg”) and connections: ”East” is ”Control Room”, ”Up” is ”Treasure Room”, ”North” is ”Nautiloid”.

There is a room named ”Control Room” with description: ”All you can notice is the mass of veins and flesh-mass that is undoubtedly the ship’s control panel.” with objects: (object named ”Control Panel” with description: ”The control panel beckons to you.” and interactions: interaction named ”Touch Button” has effect ”game win”) and connections: ”West” is ”Causeway”.

There is a room named ”Treasure Room” with description: ”Bones litter the floor alongside a scattering of weapons and armor. Mindflayer’s have no use for mortal tools.” with objects: (”Nonplayable” character named ”Mind Pet” with stats: hp = 50, mp = 0, atk = 10, def = 10, matk = 20, mdef = 20, spd = 10 and abilities:) and connections: ”Down” is ”Causeway”.

Expected Output:
The nautiloid destroys all conceptions of engineering and construction that you understand. It seems like you're standing on a living, breathing creature. Every surface pulsates and pus oozes underfoot.
>

Example 2. To run, type 'dotnet run castle.rpg' in the console.

There is a room named "Castle Gate" with description: "The castle has been taken over by a group of marauders! Retake your kingdom at any cost! The castle gate lies open to the north." with objects: ("Playable" character named "Ser Samantha" with stats: hp = 100, mp = 100, atk = 50, def = 50, matk = 25, mdef = 50, spd = 75 and abilities:) and connections: "North" is "Courtyard".

There is a room named "Courtyard" with description: "The courtyard is littered with the remains of a great battle. There is a path in every cardinal direction." with objects: ("Nonplayable" character named "Raider Cliff" with stats: hp = 50, mp = 0, atk = 50, def = 25, matk = 0, mdef = 0, spd = 50 and abilities:), ("Nonplayable" character named "Raider Shaman Lernie" with stats: hp = 25, mp = 100, atk = 0, def = 25, matk = 75, mdef = 50, spd = 40 and abilities:) and connections: "North" is "Throne Room", "East" is "Training Room", "West" is "Library", "South" is "Castle Gate".

There is a room named "Library" with description: "The quiet library's books remain untouched by the attackers. The path to the courtyard lies east." with objects: ("Nonplayable" character named "The Librarian" with stats: hp = 50, mp = 150, atk = 25, def = 50, matk = 100, mdef = 100, spd = 10 and abilities:) and connections: "East" is "Courtyard".

There is a room named "Training Room" with description: "The training room, usually stockpiled with weapons, has been looted by the raiders. To the west is the courtyard." with objects: ("Nonplayable" character named "Weapons Master" with stats: hp = 75, mp = 0, atk = 100, def = 75, matk = 0, mdef = 0, spd = 100 and abilities:) and connections: "West" is "Courtyard".

There is a room named "Throne Room" with description: "The ornate throne room is marred by the aftermath of the royal guard's last stand." with objects: ("Nonplayable" character named "Raider Chieftan Karla" with stats: hp = 100, mp = 50, atk = 75, def = 50, matk = 50, mdef = 50, spd = 75 and abilities:), (object named "Royal Scepter" with description: "The symbol of your kingdom's strength and prosperity." and interactions: interaction named "pick up" has effect "game win") and connections: "South" is "Courtyard".

Expected Output:
The castle has been taken over by a group of marauders! Retake your kingdom at any cost!
>

Example 3. To run, type 'dotnet run school.rpg' in the console.

There is a room named "Principle's Office" with description: "The infamous principle's office lies before you. It is bare of personality and reeks of shattered dreams. Salvation lies behind you." with objects: ("Nonplayable" character named "The Principle" with stats: hp = 200, mp = 200, atk = 50, def = 25, matk = 0, mdef = 0, spd = 75 and abilities:) and connections: "Behind" is "Administration".

There is a room named "Administration" with description: "A place of polarizing emotion - either the joy of being check out early for dentist's appointments, or the terror of awaiting disciplinary action. To the left is the Principle's office. To the right is the hallway." with objects: and connections: "Left" is "Principle's Office", "Right" is "Hallway".

There is a room named "Hallway" with description: "The hall is filled with lockers. Not a soul lies in sight. To your left is administration, in front of you is the cafeteria, and behind you a classroom." with objects: and connections: "Left" is "Administration", "Forward" is "Cafeteria", "Behind" is "Classroom".

There is a room named "Cafeteria" with description: "The food isn't that bad once you eat enough meals. The window

you always stare out of beckons to you." with objects: (object named "window" with description: "The window you always look at as you eat lunch seems to be slightly ajar." and interactions: interaction named "open window" has effect "game win") and connections: "Behind" is "Hallway".

There is a room named "Classroom" with description: "Your homeroom is empty. An unfinished worksheet with your name lies before you. In front of you is the classroom exit." with objects: ("Playable" character named "You" with stats: hp = 10, mp = 200, atk = 5, def = 25, matk = 0, mdef = 0, spd = 5 and abilities:) and connections: "Forward" is "Hallway".

Expected Output:
Your homeroom is empty. An unfinished worksheet with your name lies before you. In front of you is the classroom exit.
>

0.4 Language Concepts

The language doesn't actually have many moving parts - the set of primitives I want combine nicely, and the most of the functionality/computation is hidden from the programmer. As such, the programmer simply needs ideas to fill out the game. Primitives include type, name, stats, ability, and effect. Using these building blocks, we can create characters, rooms, and objects, which is essntially all that we need to build the game and the world.

The most important thing about the language is that most of the functionality/computation is hidden from the programmer. As such, the programmer simply needs ideas to fill out the game. Creating the map, executing combat, etc. are all things that the programmer doesn't consider - rather, they create the guidelines that the functionality of the language operates within. The most important things the programmer has to understand are how rooms are connected, what effects are avaialbe, and how stats interact in and outside of combat. There is plenty of room for error on the programmer's part - they could easily create an unplayable game or a game that makes no logical sense.

0.5 Formal Syntax/Grammar

```
<map>          ::= <room>
                 | <room><map>
<sentence>     ::= (<character>)
                 | (<object>)
<room>         ::= There is a room <name> <descriptor> <objects> <connections>
<objects>      ::= <sentence>
                 | <sentence>, <sentence>
<object>       ::= object <name> <descriptor> <interactions>
<character>    ::= <type> character <name> <stats> <abilities>
<type>         ::= "Playable"
                 | "Nonplayable"
<name>         ::= named <string>
<descriptor>   ::= <string>
<stats>        ::= with stats: hp = <int>, mp = <int>, atk = <int>, def = <int>, matk = <int
    mdef = <int>, spd = <int>
<abilities>    ::= and abilities: <ability>,
                 | <ability>,
                 | <ability>, <abilities>
<ability>      ::= ability <name> <effect>
<effect>       ::= has effect <string>
<connections>  ::= <connection>
                 | <connection>, <connections>
<connection>   ::= <string> is <string>
```

```
<interactions>::= <interaction>
                  <interaction>, <interactions>
<interaction> ::= interaction <named> <effect>
<int>         ::= *an int in F#*
<string>      ::= *a string in F# surrounded by quotes*
```

0.6 Semantics

<map> = The map consists of a list of rooms that the player can navigate through.

<room> = Rooms are the first step of interaction for the player. They have a name, description, list of objects, and list of connections. When not in combat, rooms are where the player exists.

<object> = An object is something that exists in a room that the player can interact with. It has a name, description, and a list of possible interactions.

<character> = The character is the vessel for both the user to play the game and the NPCS that will populate the game world. Characters consist of a type, name, list of stats, and list of abilities.

<type> = Denotes whether a character is Playable or Nonplayable. There should only be one playable character, which the user will inhabit. There can be as many nonplayable characters as the programmer desires.

<name> = Names the room, object, or character the name is attached to.

<descriptor> = A string that provides context to the room or object.

<stats> = A list of properties with int weights that determine character properties in combat.

<abilities> = A list of abilities a character possesses to use in combat.

<ability> = To be implemented. A special move that a character can use in combat. It has a name and an effect.

<effect> = To be implemented. An effect somehow changes the gamestate of the world, whether that be in combat or in the world. The only current effect is "win game", which ends the game. Effects would exist as a library of strings that are connected to a F# function.

<connections> = A list of connections that a room has to other rooms. This allows the user to travel from room to room.

<connection> = A connection denotes the direction and existing room in that direction, allowing a user to travel in that direction to a different froom.

<interactions> = A list of interactions that can be done with an object.

<interaction> = An interaction has a name and an effect, and gives the user the ability to perform some type of action on an object.

<int> = A typical int in F#.

```
<string> = A string of any length in F#, surrounded by quotes. These appear literally
in the game itself.
```

0.7 Remaining Work

There is a lot of implementation I would like to see. The most glaring feature that's lacking is the effect feature, which directly affects both abilities and objects. I would love to add effects like poison, buffs, debuffs, special damage, etc in the combat department, and outside of combat allow stuff like unlocking doors or solving puzzles to be effects. I didn't implement effects due to a lack of time - implementing what's already in the project took a lot of time and effort. Other things to add would be an exp/leveling system to increase stats, inventory system to pick up objects like weapons or armor, and other typical RPG features. It also isn't the easiest debug due to the density of words, so implementing a better coding experience would also be nice.