GRIMM: the choose your own story language

Mike Lenner, Bill Liu, Mariya Nomanbhoy, Becky Plummer

CS4115: Programming Languages and Translators

09/28/04

1. Introduction

The programming language GRIMM is designed to allow the user to easily create a "choose your own adventure" story for children. Although simple, the language is flexible and powerful enough to create complex, text-based adventure games. GRIMM is named in honor of the brothers Grimm, a pair of famous authors of children's stories.

Using GRIMM, the programmer is able to quickly and naturally create a story with characters, places, items, and choices for the player using the programmers own images and sounds. GRIMM allows the programmer to create a personalized story with an ending chosen by the child. The experience is interactive, making it much more engaging than reading a bedtime story. The child will have choices, and therefore will direct the progression of the story until they reach their own, unique ending. The language is designed so that less technical users, such as parents and teachers, are able to quickly create programs and detailed stories even if they lack extensive programming experience.

2. Motivations

GRIMM came out of our interest in creating a children's storytelling language, text-based game creation language, and FSM modeling language. We wanted to create a language that interested us and also allowed the programmers to create something unique, providing entertainment or aid to the end user of the application. Combining these interests, it is easy to see how we arrived at a "choose your own adventure" storybook language.

After researching the programming languages created by groups from previous years, we decided we wanted to create a language that allows the programmer to do something that is not already implemented in an existing language. Our language shares some features of the eMuse language. eMuse was designed to allow the programmer to create a screenplay and visualize it before getting actors involved. However, GRIMM allows the user to directly interact with the story in an open-ended, non-deterministic environment.

Lastly, we wanted a language that would allow the user of a created application to be entertained or educated in some way. Initially, we considered implementing a tool that would help students in school by using our language to visualize a problem. However, the creation of a story would be entertaining for a child and inspire them to create their own stories. Therefore, this meets our two-fold objective: to entertain the child and expose them to programming while advancing their education by facilitating them to write their own stories.

3. Goals

Inherent in the programming of all text based, role playing games are a number of common concepts. These include creating characters, items, and a space in which these characters and items exist. Using a non-domain specific language, such tasks are non-trivial. Our goal was to create a language with these constructs built in, allowing the programmer to concentrate on the creative aspects of building their adventure.

By taking advantage of the common *software* aspects of these items and places, GRIMM requires the developer to merely specify a few details for each new entity they wish to introduce in their story. So, instead of wasting time writing code to implement a front door, the developer can create such an object quickly, then move on to determine what happens when the character passes through.

GRIMM takes the developers focus away from the programming details, and puts it back to where it should lie, in the story. By doing so, the programmer can create a much richer and more intricate plot, making the adventure much more appealing. Additionally, GRIMM gives those unfamiliar with the aspects of programming and easy interface to be able to create the stories they may dream up.

4. Features

Grimm: A simple, intuitive, flexible, object oriented, interactive, multimedia, fun language.

4.1 Simple

In order for teachers and parents to use the language it must be simple for them to learn and use. These types of people are busy enough without having to undertake learning a programming language, especially if they are not studied in computer science. Therefore we wanted to create a language that would be easy for non-technical people to understand and use. We are offering these people a language that does the image processing and placement, state creation and movement, and choice implementation and execution without requiring the writer to know about the intricacies of programming. The functions are straightforward and easy to recall and use. The language is also small and basic. The programmer can make the stories as simple or complex as they feel comfortable with by choosing how many states to have, how many objects to use, and how many options the user will have.

4.2 Intuitive

For non-technical people to use GRIMM it must loosely resemble English to make story-board creation intuitive. Languages such as C++ and Java are hard for non-technical people to understand because the syntax is unfamiliar. With GRIMM, users will understand the keywords and control structures because they will be described

using English-based conventions. New story-board creators should be able to easily program-by-example by examining other stories. This will enable non-technical parents and teachers to quickly learn and start writing stories in GRIMM.

4.3 Flexible

There are many aspects to a story including: scenes, objects, characters, descriptions, and open-ended choices. We allow the writer to completely create their own world with our framework. The writer can assign attributes to items, such as names of characters. Additionally they can associate items or scenes with images that will be shown at runtime. This allows the writer to create stories involving their own surroundings as well as make-believe items using basic predefined data types such as character, scene, and object. This flexibility allows there to be an endless number of possible stories with GRIMM.

4.4 Object Oriented

GRIMM has many objects, each with their own specific attributes. There are various basic data types that are used to describe a story. One example is a "character". Each character must be created and then assigned values to its attributes such as name and gender.

We thought initially that non-technical people could be confused by the idea of Object Oriented, but we feel that grouping all aspects of a data type will instead be helpful to them due to its clear organization. For example, all of the attributes of the character "Garfield" are directly attached to the character instance:

```
Character Garfield
{
    Garfield's name is "Garfield"
    Garfield's gender is "male"
    Garfield's picture is "garfield.jpg"
    Garfield's age is "12"
}
```

Object Orientation will help the writer organize objects and their attributes and therefore facilitate the creation of a story.

4.5 Interactive

Once a GRIMM program is created and presented to a child the possibilities are endless. The child has several choices in every scene to perform actions such as going to another scene, manipulating items, and interacting with other characters. Here are some examples of actions that can be performed:

```
wear apron
```

```
get food
goto kitchen
eat food
open door
tell garfield hello
buy catfood
```

The open-ended world is a robust feature that makes GRIMM so enjoyable for children. The child's interest will be captured while they are working within the story. Aside from the child being able to make choices, the child gets to decide how the story progresses and ends based on the choices that they make.

4.6 Multimedia

A special feature of GRIMM is the ability for the programmer to input any images of their creation into the story. Each scene in the story has the ability to display a visual element. This allows the story to become more personal for the child who is exploring through the world. As the character moves throughout various scenes, images will be displayed as a background. The writer also has the ability to add musical events to the story such as a sound effect or musical score. We feel that the multimedia capabilities of GRIMM will help the writer and user to enjoy the experience.

4.7 Fun

GRIMM allows the writer to create any story with an unlimited number of possibilities. We want people to have fun with the flexibility, interactivity, and multimedia aspects in order to engage a child in an interesting story. Both aspects of the program should be enjoyable: development and usage. We want the writer to be engaged in the development by making the language intuitive and simple. Its flexibility allows the writer to be creative. The child will become engaged because of the engaging nature of the stories and perhaps partake in the development of new ones.

5 Summary

In the GRIMM language, we have opened up programming to the programming novice, taking away the intimidating syntax and replacing it with intuitive, domain specific keywords. We have also provided a flexible tool, such that games and stories can be created as simple or as complex as the developer desires. GRIMM also brings a new education tool to the classroom or the home, one in which the learned skills translate directly into an adventure both teacher and student can enjoy. And finally, GRIMM stops the wasted time spent by previous developers in coding the same basic concepts for every game or story they create.

However, in spite of all these advantages, the most important aspect of the GRIMM language is that of allowing the programmer to focus on what is important. GRIMM redistributes the programmer's effort to where it belongs - on the story, not on the code.