# Design and Development of a Virtual Dashboard for a Polar Robot


## Author: DeQuincy Faulcon

### Mentor: Dr. Arvin Agah


## I. ABSTRACT


The goal of this research project is to design and develop a graphical user interface (GUI) for the MARVIN II polar rover that will be accessible over the Internet. This interface will resemble a simulation of the rover in the format similar to such video games as Test Drive 2 for the Sega Genesis and F-Zero GP Legend for the Gameboy Advance. Before programming, a conceptual diagram has to be created. In order to accomplish this, the main source for the inspiration for the project had to be used. There are some features on the conceptual layout that have not yet been applied to the rover itself, but possibly may be included in the future. As the capabilities of the rover improve, the abilities of the virtual dashboard should improve as well.


## II. INTRODUCTION

The purpose of this graphical user interface is to provide anyone with a view of what the rover is doing at the time from the first-person perspective and to output various data that are available. The first person perspective simply is one where the one is allowed to have a view as if one was in the vehicle itself. Such data that is being outputted but not limited to: pitch, yaw, roll, speed, fuel and the GPS position. The Java programming language was created in the year 1994 and is incorporated into nearly every Web-based program. The virtual dashboard will load in from the website in the form of an applet. There were many factors involved on making the decision to use the Java programming language.

The most obvious reason for Java's popularity is that it can run on PCs, Sun workstations and Macintoshes. Another reason is because the features Java contains are borrowed from C and C++, making it somewhat familiar [1]. Xerox's Palo Alto Research Center (PARC) created the first GUI. The first GUI was then enhanced by Apple Computer and then further enhanced by Microsoft. GUIs were well received because they were able to allow almost anyone learn how to use a program within a fraction of the time [2]. Also, Microsoft Paint was used in the creation of the conceptual design because of its simplicity.

## III. METHODS

Before the actual GUI is created, a few conceptual models have to be created. In order to begin the design process, the initial inspiration had to be reviewed. The source for this project was video games. After careful deliberation, it was decided that simulation and racing-type video games were to be used as a template. The popular arcade racing game, Daytona Racing, was used because it had a simple, but effective layout that was informative and easy to read from the first-person point of view. These models were created in Microsoft Paint and edited using Adobe Photoshop. From the models, one was chosen and refined

to reflect the desired functionalities of the GUI.

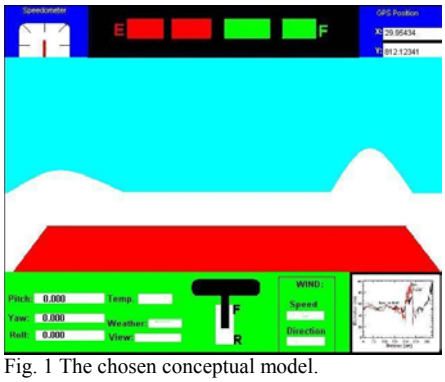

Fig. 1 The chosen conceptual model.

With the easy part of designing the desired GUI completed, the next step was to learn how to program the GUI. The programming language of choice was Java because of its portability.

Before there was any programming to be done, the Java 2 Standard Edition Software Development Kit (J2SE SDK) had to be downloaded and installed. When that was finished, the task of learning how to program in that specific language came to hand. There were many books as well as tutorials on the Internet available. To learn the basics of the language, the following books were used: Introduction to Programming Using Java and Java Demystified. The first book, Introduction to Programming Using Java served as a reference to learn the basics, such as defining classes and implementing loops. The other book, Java Demystified went into more detail about applets applications. Originally, in order to write, compile and run Java code, two different programs were used. First, the code had to be written in a text editor, such as Notepad and then compiled and ran using Microsoft DOS command prompt window. Finding this process to be time-consuming, some Internet exploration was conducted to find a better, more efficient way to code, compile and run Java programs. After performing a search using a search engine, many Websites were displayed with the term, integrated development environments (IDE). IDEs are programs that streamline the process of editing, compiling and executing code by allowing all three to be performed within the application [1]. As a result of the Internet searches, it was decided that Textpad would be used.

Helios Software Solutions developed the Textpad IDE. Textpad has the ability to use various compilers as well as to be able to display both Java applications and applets. This program was very useful as a beginner application for using the basics, but using a linear set-up to create applets was both time-consuming and inefficient. This meant a return to the Internet to find an object-oriented IDE.

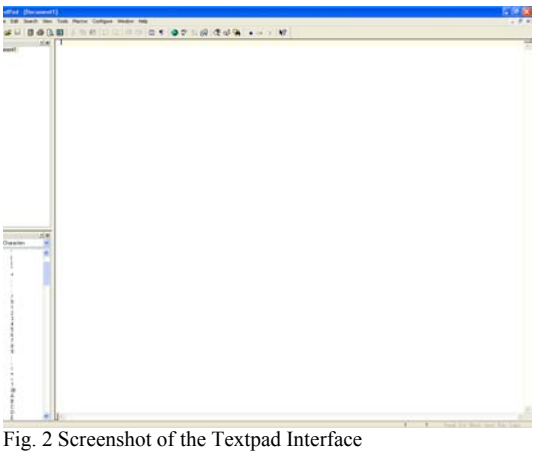
Fig. 2 Screenshot of the Textpad Interface

Object-oriented IDE's are more practical in these situations because of the ability the user has to design the graphical user interface to meet the desired output of the user. The first object-oriented IDE to be used was JFrame Builder.

Mars Microsystem Company developed the JFrame Builder. This IDE was very limited in its capabilities,

specifically in design options. JFrame Builder could only generate code for the Swing libraries and it could not handle images. It is more desirable to have the code based on the AWT libraries because Swing is relatively new and may not be supported on some of the older web-browsers.

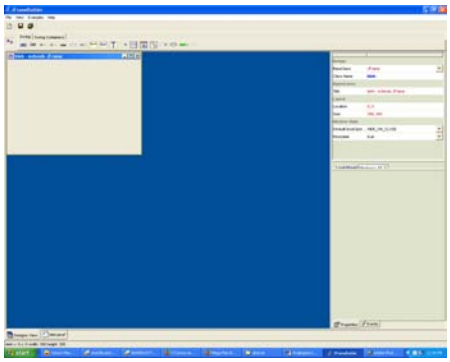

Fig3. Screenshot of JFrameBuilder

With JFrame Builder being unable to meet the needs of the project, a replacement program was found. The next program to be used was Borland's JBuilder X.

JBuilder X was a vast improvement over JFrame Builder. J Builder X was created and developed by the Borland Software Corporation. This program allows for the creation of applications and applets using either Swing or AWT libraries. The layout of JBuilder X is easier to understand and allows the user to work on more than one part of the main program at a time. Even with all of its features and functions, JBuilder X had the same problem as JFrame Builder. This problem dealt with the lack of ease to be able to load pre-made images. To rectify this problem, another IDE was found by the name of Eclipse.

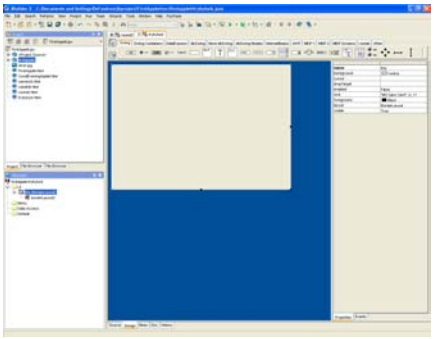

Fig 4. Screenshot of JBuilder X.

Eclipse is a multiple language IDE, meaning that it can compile more than one type of programming language. This IDE is part of an open source project, meaning anyone can contribute to making it better through plug-ins. A plug-in is a supplementary program that is created outside of the main program. Its purpose is to enhance what is already there. Once Eclipse was downloaded and installed, it was found that a plug-in was needed in order to use an object-oriented approach to create an applet. After looking through various plug-ins, Jigloo was found to be the most useful. This plug-in was created by Cloud Garden. With the plug-in installed, the GUI was completed with the necessary images.

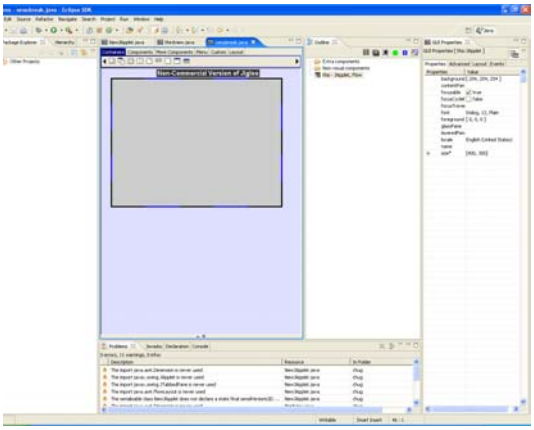

Fig 5. Screenshot of Eclipse with Jigloo.

## IV. **RESULTS**

The result of this project is a GUI that provides the first person view from the rover. From this view, the user can determine many things. Some of the information the viewer can see is: pitch, temperature, fuel, and location. The GUI itself can be compiled and executed. Being that the design GUI has just been recently created, it will take some time before it is able to receive data. The construction of this GUI is only the beginning of its use.

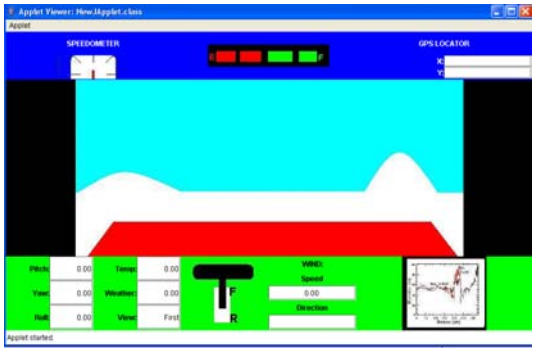

Fig 5. Actual Applet of the GUI

## V. Conclusion

In conclusion, the design and implementation of the GUI itself can be done. With this success, the project can be built upon and expanded in seemingly limitless ways. This project itself gave the impression of the creation of a video game, which means that there are always ways to improve upon the existing product. One possibility is to add other views, such as third person and bird's eye views. Another possibility to enhance this project is to create an interchangeable dashboard to suit the taste of the user. A third is to create, edit, and upload pictures to give a better impression that the rover is in motion. In order to use the actual code however, a licensing fee must be paid to Cloud Garden. This initial GUI is only the beginning of what can possibly be done.

## VI. REFERENCES

[1] Arnow, David M. Introduction to Programming Using Java. Reading: Addison Wesley, 2000.
[2] Keogh, Jim. Java Demystified. New York: McGraw-Hill/Osborne, 2004.
[3] Nourie, Dana. "Building an Application ." Online Training and Tutorials:Graphical User Interfaces and Printing. Nov 2001. Sun Developer Network. 20 june 2005 <http://java.sun.com/developer/onlineTraining/>.
[4] Tutorial:Building an Applet. 16 Sept 2002. 27 June 2005 <http://www.ida.liu.se/~TDDB62/introjb/firstapplet/firstapplet_part1.html#createapplet>.