© 2017 IJSRSET | Volume 3 | Issue 1 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Multi Graphical User Interface Compiler

Harshad Rane, Brijeshkumar Gupta, Akshay Mhatre, Yogesh Gaikwad

Padmabhushan Vasantdada Patil Pratishtan's College Of Engineering, Sion, Mumbai, Maharashtra, India

ABSTRACT

Today it is difficult to design a good GUI in a widely acceptable language. We plan to design simple user interface language with easy to understand constructs for designing user interface. But of course the user will not like only the GUI to be in the language in which he is developing an application. To overcome this issue we plan to implement a compiler, to be written in Java, which will combine this new language to a target language such as Java. Thus a user will get the code for the GUI he is designing in a high level language. We also plan to provide an IDE for writing the new language and for compiling it to the target language. The system consists of a compiler for compiling and translating JUICE SCRIPT into specified target language, which may be Java Swing, Java AWT, XUL or HTML. The system must be expandable for inclusion of new languages also. An IDE for developing the JUICE SCRIPT is to be provided for easier development of the JUICE SCRIPT.

Keywords : JUICE, Multitargeted compiler, JavaCC, XUL

I. INTRODUCTION

To work with a system, the users need to be able to control the system and access the state of the system. User can interact with system with the help of graphical user interfaces (GUI) which accept input via devices such as computer keyboard and mouse and provide graphical output on the computer monitor. The graphical user interface is a computer interface that uses graphic icons and controls in addition to text. The user of the computer utilizes a pointing device, like a mouse, to manipulate these icons and controls. This is considerably different from the command line interface (CLI) in which the user types a series of text commands to the computer. Today designing a good GUI in a widely acceptable language is not an easy task.

Our system consists of a compiler for compiling and translating JUICE script into specified target language, which is Java Swing, Java AWT, XUL or HTML. The system must be expandable for inclusion of new languages also. An IDE for developing the JUICE script is to be provided for easier development of the JUICE script.

II. METHODS AND MATERIAL

Here we have mentioned the existing tools for our system briefly.

1. Jvider - It is the GUI builder tool for java swing applications. With Jvider you can easily design the graphical user interfaces for your java applets and applications. It provides understandable interface, draggable and resizable components. It has ability to generate the source code in java language. It has ability to export the source code as frame or applet. It is platform independent and runs on all environments that support Java Virtual Machine (JVM).

Disadvantages:

- It is used for creating GUI in one language only.

2. GrafiXML - It is software used to generate graphical user interface (GUI) and save them in a UsiXML format language. GrafiXML basically work as other GUI Builder but also contains tools to localize your applications.

III. RESULTS AND DISCUSSION

A. System Working

All the code for this system is segregated into different Class files. An IDE is developed where user can enter code in JUICE script and choose an option for target language.

The compiler has been implemented using JavaCC parser generator. It detects all the tokens described in JUICE Script. It also checks for parsing errors. Each GUI designing language has some common GUI components. For each component class files are created. All the Classes pertaining to a given target language are similar to Classes of other target language.

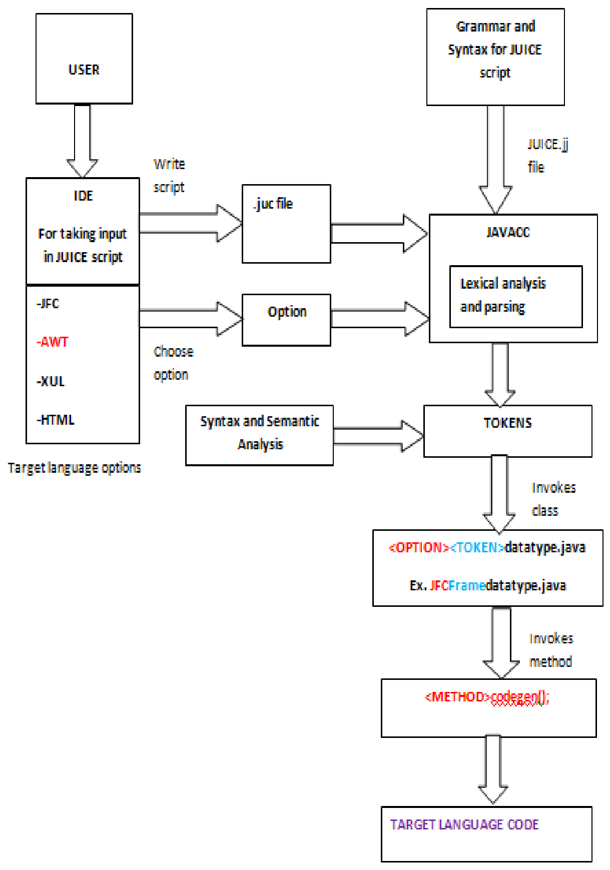

Figure 1 : System Architecture

The compiler takes input file with .juc extension and converts it to target language based on the switch entered. It references the class related to a token for a given target language by adding the switch for language before the common class name for that token.

For example, on getting token 'button' and switch AWT the class referenced is AWT ButtonData Type. Then a call is made to a function written in the referenced class based on which property or event is mentioned for that control. The code for the same is written in the output file by the called function. This Design Strategy makes JUICE scalable i.e. adding new language is easier. Error detection and Maintenance is also easier.

B. Proposed Algorithm

Step 1. The user has to open the IDE provided.

Step 2. The user has to write input script and save the file with .juc extension.

Step 3. The file is passed as an input to the parser.

Step 4. The parser parse's the tokens one by one and pass it to the switching mechanism.

Step 5. The user has to select target the through command switch.

Step 6. The tokens and command switch is passed as an input to the switching mechanism.

Step 7. Switching Mechanism generates the class file name by following formula: Command Switch + Token + Data Type.

Step 8. According to the command switch the class will be passed towards the code generator.

Step 9. In the code generated the component will be selected as per the class name generated by switching mechanism.

Step 10. As per the sub tokens generator by the parser the methods in the component are selected

Step 11. The methods invoked will be writing the GUI code in the output file.

Step 12. The output file is compiled to get the GUI in desired language

IV. CONCLUSION

This paper shows that JUICE is a rapidly evolving area of research and development. We discussed only the key problems in this area and presented some known solutions. One key research problem that we still face today is the development of truly easy, less complex, and time saving techniques for generating GUI in different languages.

V. REFERENCES

[1]. https://javacc.dev.java.net
[2]. http://www.scifac.ru.ac.za/compilers/conts.html
[3]. http://java.sun.com/products/jfc/download.html
[4]. http://downloadl-lnw.oracle.com/javase/1.4.2/docs/api/java/awt/package-summary.html
[5]. https://developer.mozilla.org/en/introduction_to_xul.
[6]. http://download.oracle.com/javase/tutorial/reflect/index.html

[7]. http://java.net: JavaCC tm]: Grammar Files
[8]. Alfred Aho and Jeffrey D. Ullman,(1986) Compilers: Principles of Compiler Design.
[9]. Advance programming in java NIIT, Prentice Hall of India, ISBN-81-203-2415-3
[10]. Raphael A. Finkel, Advanced Programming Language Design.
[11]. Andrew Appel, Jens Palsberg: Modern Compiler Implementation in Java. Cambridge University Press, 2nd edition, 2003.
[12]. Benjamin Michotte, Jean Vanderdonckt, “GrafiXML,AMultitarget User Interface Builder based on UsiXML”.