METU, Department of Computer Engineering
Graduation Project
Proposal Form

Project Information

Title

Multi-purpose FlowChart Interpreter

Target

Public [ ] Restricted [✓]

Proposer Information

Name(s)	Mehmet AKALIN Ertuğ UFUK Elif ŞAHİN Merve Gizem ŞENEL
E-Mail(s)	*mehmet.akalin@ceng.metu.edu.tr*

IP (Intellectual Property) Information

Project Description and Background Information

Description

Our project, MultiPurpose FlowChart Interpreter, converts FlowChart to some popular programming languages such as Python, C++ and allows users to track flow of converted code on FlowChart simultaneously besides showing profiling of code again on FlowChart. Also our project will be able to convert source code that is provided by user to Flow Chart. In addition, since documentation of software frequently requires visualization of data structures, our application will convert data structures in C, C++ and Java into graphs. It will allow editing, publishing and simple animations of this data structures.

Similar Products/Projects

1. Raptor: Flowchart-based programming environment, designed specifically to help students visualize their algorithms.
2. coCodeX – Flow to Code: Converts basic FlowChart Diagram to Source Code.
3. Flow Chart Visual Programming Language: Flowchart interpreter that is only available in Turkish.
4. Doxygen
5. Dia scripts converting source into UML

Justification of the proposal

-The purpose of our project is to enable users to develop programs without writing code by only working on the algorithm. The analysis process will be more detailed and easy to understand.
-We develop this project since there are some issues on the development of the programs. The developers spent a lot of time on coding, syntax and code errors. Our project aims that developers spend this time on their algorithm and its tracing. There are various tools for algorithm visualization and data structure visualization. Also tools exist for UML to source code and vice versa. However most tools are create visualizations on one basic layout and form. The support for languages like C is missing. They do not have interaction or editing features.
-With this project, we want to eliminate loss of time, ambiguity of algorithm and poor coding. Moreover, our project can be used to modify visualizations after creation and exported on different formats. Also simple animations can be constructed with editing frames.

Contributions, Innovation and Originality Aspects of the Project

There is no visual tracking of debugging, visual performance analysis and multi-language support in any existing similar programs. Also existing similar programs don't allow users to define functions, structers and classes, importing standart libraries and OOP Design. In addition to this, any similar program doesn't convert standart source code to FlowChart. Lastly, all similar programs have insufficient, inefficient and difficult interface to learn.

In light of this lacking abilities, we bring following innovations to our project relatively current similar projects:
-Visual Debugging
-Visual Profiling Tracking (by coloring statement depending on their performance with respect to others [red, green])
-Detailed performance information of statements (How much time has been spent on which statement, how much RAM is used for which statement etc.)
-Converting standart source code to FlowChart
-User-friendly interface
-Multi-language support
-Ability to define function blocks on FlowChart and Recursive/Nonrecursive usage
- Ability to import standart most used libraries
-Interactive tutorials for beginners or willing users to learn about algorithms (By forcing users to do something, Drag&Drop etc.)
-Ability to define classes and in light of this ability to work Object-Oriented Programming

Technical Aspects of the Project

Our project is a Web Application that is going to be developed in C++, JavaScript and Python. C++ and Python are going to be used for headers and functions in server side to parse code and give outputs that are going to be visualized in Browser-side functions using JavaScript Headers like joint.js. Editing, exporting on SVG and other vector formats, saving and loading the flowcharts will be available. For the development tutorials, frame by frame animation support will be included.

Targeted Output, Targeted User/Domain Profile

Our project aims to create an IDE which will allow software developers to focus more on algorithms instead of coding, syntax etc. Users will also be able to have a visualisation of their written code and its execution process
Anyone who is willing to produce algorithms without coding or to trace their code via a generated flow-chart.

Project Development Environment

A C-like compiler will be used for parsing flowcharts on the server side.
Browser side libraries like jquery and/or angularjs, nodejs.
Visualization libraries like D3.js.
HTML5, SVG.
Extra libraries and plug-ins will be searched later.

External Support

Some compilers that are going to used for profiling are going to be embedded to our project.

References

1. Raptor (http://raptor.martincarlisle.com/)
2. coCodeX – Flow to Code (http://www.cocodex.com/flowchart-to-code/index.html)
3. Flow Chart Visual Programming Language (http://daltinkurt.com/Yazi/117/Flow-Chart-Visual-Programming-Language.aspx)
4. https://www.cs.usfca.edu/~galles/visualization/Algorithms.html
5. http://data-structure-explorer.herokuapp.com
6. http://d3js.org/
7. http://dia2code.sourceforge.net