A Comparative Study: Reverse Engineering Flowcharting Tools

Nivedita Tiwari[1], Lalji Prasad[2]

[1]*PG Student, Truba College of Engineering & Technology,*

[2]*Professor, Truba College of Engineering & Technology*

Abstract - The paper describes the Reverse Engineering methodology used to produce from source code a hierarchy of Flow Charts at different levels of trance and ensure consistency with code. Reverse engineering is the process of analyzing, maintaining and evolving legacy systems to understand the system through analysis of its code structure, function and operation by program discernment. The Flowchart is an approach for specifying, constructing and visualizing the system graphically and has been in practical use on a very wide basis but still lacks formal and precise understanding. By Reverse Engineering it is useful in maintenance and restructuring activities during the development of software systems. This paper illustrates several Reverse engineering tools and functionality of such tools varies according to their features and capabilities. We identified benefits and shortcomings of these tools and assessed their applicability for software systems, their usability, and their extensibility as a case study.

Keywords - Reverse Engineering, Tool Evaluation, Flow Charts structure.

1. INTRODUCTION

Reverse engineering provides a new approach for requirement specification of a system [1]. Reverse Engineering tools are very useful, but not widely used because that tool is not integrated and lack of many necessary features and capabilities [2]. In addition, a Reverse Engineering process threatens many of the assets of the rejuvenate and modernize approach. This approach is always and viable if the source code for a system exists. The code generation indicates that a generator reads the code or documents related to the graphic model and generates a high-level language program, just like C, C + +, Java, Perl, Ruby, Python and HTML and so on [3]. A UML tool can convert UML graphics into a specific language to predict the performance of system [4].

In this paper, we compare the various tools on the basis of their capabilities and features that captures the functionality of the system and the design at an abstraction level that removes the need of language implementation. Our current research effort focuses on a selection of these tools that will be useful in maintaining the consistency of design document with its code and for understanding large systems.

The remainder of this paper is organized as follows. In the section 2 we describe the works that are previously done as literature survey. In the section 3 brief introduction to structural analysis is given; focus on the use of flow charts. The main part of this paper is section 4 which will present the feature analysis of supporting tools that generate the flow chart through source code.

2. LITERATURE SURVEY

There are various Reverse Engineering tools that generate flow charts through source code. It is not possible to study each and every tool [12]. According to Carlisle M, Wilson T, Humphries, J and Hadfield, M [7] Raptor is a tool for flow chart based visual programming Environments. It is an MS Windows based application that generates and perform black and white flowcharts by dragging and dropping complete structures onto the flowchart instead of individual notations and arrows. Code visual to flowchart [5] can be used to introduce its flowcharts to Microsoft Office application including Visio. This tool shows how the flowchart and the code relate to one another and is a useful indicator.

Westphal et al [8] define that without the use of diagrams or flow charts, it is difficult for beginners, even with pseudo code to communicate the flow of a program. While flow charts are not well suited for large and complex programs. Flowchart plays an important role in system analysis, preparatory design of algorithm and system maintenance aspects.. So, automatic generation of a flowchart from specific language code will be a very important practical significance, it allows the designer to design the system from high-level functions without concerning for complicated code [11]. According to the Andrew Scott [6], the use of the tool focuses on using flowcharts to develop visual solutions to basic programming problems from which syntactically

correct program code is generated. The animation features and interaction between the visual and code representations reinforce student understanding of both the visual solution and program statement flow.

3. STRUCTURED ANALYSIS

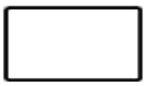
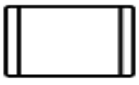
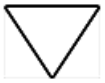
Structured Analysis defines the structural elements specifically designed to describe and understand the system. It offers following building blocks to represent flow charts:

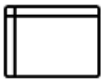
A. Symbols Used

A flowchart is a formalized graphic representation of a logic sequence, work or manufacturing process, organization chart, or similar formalized structure. The purpose of a flow chart is to provide people with a common language or reference point when dealing with a project or process. A flow chart is the visualization of a system or process consisting of five basic objects:

- Process / Operation Symbols
- Branching and Control of Flow Symbols
- File and Information Storage Symbols
- Input and Output Symbols
- Data Processing Symbols

Table-1: Flow Chart Symbols

Category	Symbol	Name	Function
Process/ Operation Symbols		Process	Represents a process, action, or function.
		Predefined /Subroutine Process	Represents Series of process flow steps that are formally defined elsewhere.
		Preparation	Represents a process flow step.
		Manual Operation	Represents a step that must be done manually.
		FlowLine/ Arrow/ Connector	Represents the direction that the process flows.
Branching and Control of Flow Symbols		Terminator	Represents the start points, end points.
		Decision	Represents a question or branch in the process flow.
		Connector	Connects separate elements across one page.
		Off-page Connector	Connects separate elements across multiple pages.
		Merge	Combines multiple paths.
		Summing Junction	Represents when multiple branches converge into a single process.
Input and Output Symbols		Data(I/O)	Represents data that are available for input or output.
		Document	Represents the input or output of a document.
		Multi-Document	Represents multiple documents or reports.
		Manual	Represents the manual

		Input	input of data into a computer, through a keyboard.
File and Information Storage Symbols		Stored Data	Represents data housed in a storage device.
		Internal Storage	Represents data stored in RAM.
Data Processing Symbols		Collate	Represents a process step that requires organizing data, information, or materials according into a standard format or arrangement .
		Sort	Represents the sorting of data, information, materials into some pre-defined order.

B. Construction of a Flow Chart

Flowcharting is a tool developed in the various fields, for showing the refined form of any process. A flowchart is a diagram made up of boxes, diamonds and other shapes, connected by arrows. Flowcharting combines symbols and flow lines, to show figuratively the operation of an algorithm.

Following are the steps for developing a Basic Flowchart:

1. Define the Process and Purpose of the Flowcharting.

2. Establish the Process Boundaries (Starting and End points).

3. Identify the Sub processes that are included in the defined Processes.

4. List the Steps and Connect in order in which they relate to each other.

5. Examine and Assign a flowchart symbol for each step.

6. Review and title the flowchart.

7. Improve the flow chart (work back to the first step).

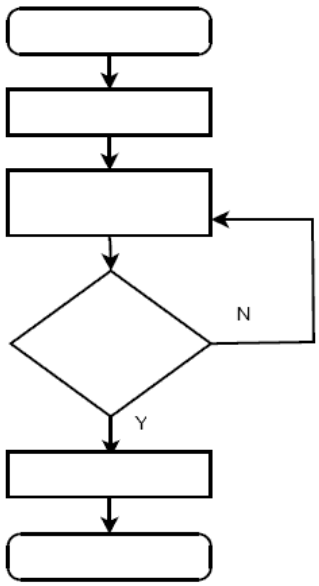


Fig.1. Basic Flow Chart Structure

C. Types and Uses of Flow Charts

Flow Charts are used in analyzing, designing, documenting and managing a process or program in various fields. Some of the common uses of flowcharts include:

Table-2: Types and Uses

Category	Types	Uses
Document Flow Charts	Basic Flow Chart	• Showing controls over a document-flow through a system. • Planning a new project. • Documenting a process
	Swim lane flow chart	
Data Flow	Data Flow	•Showing controls

Charts	Diagram	over a data-flow in a system. •Data management
System Flow Charts	EPC diagram	• Showing controls at a physical or resource level. •Modeling a business process. •Managing workflow.
	Workflow Diagram	
Program Flow Charts	Basic Flow Chart	• Showing the controls in a program within a system.. •Planning a new project. •Mapping computer algorithms. •Chemical and process engineering.
	SDL Diagram	
	Process Flow Diagram	

4. SUPPORTING TOOLS

For the evaluation, we choose six reverse engineering tools that generate flow charts significantly, each of which represents a different category of reverse engineering tools:

-Code visual to flowchart is an extensible, interactive tool for reverse engineering several programs in various languages. It is used to understand, evaluate and redocument existing code.

-Visustin v7 FlowChart generator provides an efficient and portable environment with a comfortable user interface.

-Imagix 4D is a comprehensive program understanding tool for C, C++ and JAVA programs. It is used to rapidly check and systematically study of software at high-level design.

- Code to Flowchart Converter is an open, extensible and scalable programming environment for C ,C++ and many languages which also support reverse engineering activities.

- Raptor is a public domain tool that is generated by parsing the application and allows editing these representations.

-B# is a classic reverse engineering tool provides the feature for flow chart construction in an efficient way.

We investigated the capabilities of the above reverse engineering tools. Our main focus on the tool features that generate flow charts.

A. Code Visual to Flowchart

Code to Flow chart generator is an automatic flow chart generator software. It can reverse engineering a program, create programming flowchart from code. With the aim of aiding programmers understand and document source code, CVF converts program codes into flowcharts. The flowchart view auto up-dates representing the program visually as each line of code is entered into CVF text editor [3]. CVF presents its flowcharts and code side by side which are synchronized via highlighting. Clicking a line of code will cause a relevant flowchart component to highlight and vice versa. This shows how the flowchart and code relate to one another and is a useful indicator. CVF can be used to introduce its flowcharts to Microsoft Office application, including Visio, Bmp, PNG, Word, Excel and PowerPoint flowcharts document. So it is really a useful documentation tool.

Code Visual to Flow Chart [5] is designed to visualize procedural code of any complexity. It works best with executable code, but it can also consume non-executable declarations. For the best results, consider removing the non-procedural parts, as they sometimes result in clutter in the diagram. You can visualize an entire source code file, a single class, a single procedure or just a part of it. If the code is long and the flow chart large, it often makes sense to remove some code and focus on the crucial points of the algorithm. Code Visual to Flow Chart works at the statement level. It doesn't care about the internal parts of complex statements, such as the Booleans in an if statement. It supports Visio, Word, Excel, C, C++, VC++, VB, VBA, Qbasic, VBScript, ASP, Visual C#, VB.NET, Visual J# .NET, VC++.NET, ASP.NET, Java, JSP, JavaScript, Delphi, Pascal, PowerBuilder, PHP, FoxPro, PeopleSoft SQR, PL/SQL, T-SQL, Perl.

B. Visustin v7 FlowChart generator

Visustin is a flowcharting software that Visualize and reverse engineers your source code and converts it to flow charts automatically. Visustin reads the if and else statements, loops and jumps and builds a diagram fully automated.Visustin flowcharts ABAP, Action Script, Ada, ASP, several

assembly languages, BASIC, .bat files, C, C++, C#, Clipper, COBOL, ColdFusion, Delphi, Fortran, Java, JavaScript, JCL (MVS), Java Server Page and many more [15].

The charts visualize your code and optionally the comments as well. Use flow graphs in your project documentation. Save files in BMP, GIF, JPEG, PNG, TIFF, MHT, EMF or WMF format. Create web pages showing both the code and the flowchart.

C. Imagix4D

Imagix 4D helps software developers comprehend complex or legacy C, C++ and Java source code [10]. By using Imagix 4D to reverse engineer and analyse your code, you are able to speed your development, enhancement, reuse, and testing.
You can visually explore a wide range of aspects about your software - control structures, data usage, and inheritance. All these are based on Imagix 4D's precise static analysis of your source code. Learn unfamiliar code. Eliminate bugs due to faulty understanding. Get new hires on board faster. Spend time engineering, not reading through code. You can visually explore a wide range of aspects about your software -control structures, data usage, and inheritance. All based on Imagix 4D's precise static analysis of your source code.

The Flow Charts provide visualization of the control flow, or program logic, occurring inside the functions in your software. For complex functions that may consist of hundreds of lines of source code, these diagrams can help you more quickly grasp the internal logic of the routine. A Flow Chart diagram presents the complete internal logic, of a single function at a time. For the next level higher level of abstraction, multiple functions, but with reduced internal logic use the Control Flow Graphs.

D. Code to Flowchart Converter

Code to Flow Chart Converter can automatically generate flowchart/NS chart from source code [14]. By visualizing the source code, it makes you understand it better. It is also a very useful tool for program developers and documentation writers. By using Code to Flow Chart Converter [9], programmers can easy access to get project overview, during the process of programming, programmers can preview their changed code structures at all times. It will be a great help for them to add new ideas into their source code. Code to Flow Chart Converter presents the source code in a straightforward way. It provides document writers with a visual diagram / flowchart. Those flowcharts make a document writer a very clear mind which is very crucial to their writing.

Code to Flow Chart Converter supports to fully expand the flowcharts, even when there are many rows of source code. All your codes can be completely displayed in the large flowchart. It supports syntax highlighting of source code, it means, when you click any logical box on the flow chart, the corresponding source code will be highlighted, it helps you to clearly view the program, and it helps the programmers to find out errors in an easier way. Code to Flow Chart Converter supports to export flowchart or NS charts to Visio, MS Word, XML, SVG, BMP.

E. Raptor

This tool was expanded for the United State Air force Academy an introduction and computing courses. The main goal of Raptor is improving problem solving skill of novices as well as shunning the syntax errors [7]. This tool tends to generate and perform black and white flowcharts by dragging and dropping complete structures onto the flowchart. This tool maintains the procedures and arrays as well as suggested a library of inbuilt procedures to execute IO with text files and define the data types of variables. As considering the focus area, the Raptor has the advantage of converting its flowchart into code, maintaining the Ada, C++, C# and Java. It must be highlighted though, variables and arrays. The program is not initialized, which could lead to syntactically incorrect code and not executable without important manual modification in an external environment.

In addition, the novice programmers would face with specific problems related to the flowchart to the respective codes. Moreover, it would also not be able to maintain the development of program testing, debugging as well as tracing skills. It is worth mentioning, lack of some significant characteristics like code viewing and execution, and syntactic omissions would counteract the code generations specification of Raptor [18].

F. B #

B# is a tool based on MS Windows based application. Programmers can use B# to construct a structured flowchart and in this tool code would be generated from the relevant flowchart in Pascal [17]. As items are added, the programmers determine related parameters which are syntactically checked, and any errors are shown to the

programmers. This tool also offers programmers to see the correlation between a code and flowchart which is side by side. Thus, synchronizing would be one of the prominent features of B#. When programmers choose the component of flowchart, the relevant line of code will be highlighted. In addition, this tool offers a visualized program execution by highlighting every single component of the flowchart and respective line of code as it is executed. The flowchart notation of this tool is not matched with standard flowcharting rules; which means B# is not in line with the flowcharting notation normally utilized in textbooks. Hence, B# will not be a transferable program design methodology and at the start it ought to be involved with extra learning [17].

Tool Selection Criteria

Table 3 shows features of the tools. The selected tools should also be either under active current development or be related to scientific publications of software maintenance.

CVF: Code Visual to Flow chart

V7: Visustin

CFC: Code to Flow chart generator

I4D: Imagix 4D

RTR: Raptor

B#

NA: Not Applicable

Table-3: Tools Comparison

TOOLS → / FEATURES ↓	CVF [5]	V7 [15]	CFC [9]	I4D [10]	RTR [7, 18]	B # [17]
Flowchart code automatically	Y	Y	Y	Y	Y	Y
STRUCTURAL RULES ENFORSED	Y	N	N	Y	Y	N
LANGUAGE INDEPENDENT	N	N	Y	Y	Y	Y
REVERSE ENGINEERING CONCEPT	Y	Y	Y	Y	Y	Y
HIGHLIGHT CAPABILITY OF SOURCE CODE	Y	N	N	Y	N	Y
OS Dependency	N	Y	Y	Y	Y	Y
CODE REVIEW	Y	Y	N	Y	N	N
BULK CHARTING	N	Y	N	N	N	N
COMBINE DATA AND DIAGRAM	Y	N	N	Y	N	N
RESTRUCTURE	Y	Y	N	Y	N	N
MULTI-PAGE PRINT	Y	Y	N	Y	N	N
EDIT FLOW CHART	Y	Y	N	Y	Y	Y
ERROR FEEDBACK	N	Y	Y	Y	Y	N
STATIC VIEW	Y	Y	Y	Y	Y	Y
PERFORMANCE	HIGH	HIGH	HIGH	HIGH	HIGH	HIGH
CLARITY	HIGH	HIGH	HIGH	HIGH	HIGH	HIGH
PUBLISHED BY	FATESOFT	AIVOSTO	COCODEX	IMAGIX CORPORATION	NA	NA

REFERENCES

[1] E. J. Chikofsky, Reverse engineering and design recovery: A taxonomy. *IEEE Software*, pages 13–17, Jan. 1990.

[2] Muller H.A. Wong K., Tilley S.R., Reverse Engineering: A road map, proceedings of conference on the future of software engineering, International conference on software engineering, 2000.

[3] Xiang-Hu Wu, Ming-Cheng Qu, Zhi-Qiang Liu, Jian-Zhong Li: Research and Application of Code Automatic Generation Algorithm Based on Structured Flowchart Journal of Software Engineering and Applications, 2011, 4, 534-545 doi: 10.4236/jsea. 2011.49062.

[4] D. Harel, "State charts: A Visual Formalism for Complex Systems," Science of Computer Programming, 2007.

[5] FATESOFT (2009), Code Visual to Flowchart, FateSoft, Eden Prairie - NM - USA, [Accessed 29/ Auber, D., Melancon, G., Munzner, T. And Weiskopf, D. (2010) SolidSX: A Visual Analysis Tool for Software Maintenance. Poster Abstracts at Eurographics/ IEEE-VGTC Symposium on Visualization. [10/2009] http://www.fatesoft.com/s2f/.

[6] Andrew Scott, Mike Watkins and Duncan McPhee. , (2010), E-Learning For Novice Programmers, A Dynamic Visualization and Problem Solving Tool, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4529966.

[7] Carlisle M, Wilson T, Humphries, J and Hadfield, M, 2004, RAPTOR: Introducing Programming To Non-Majors With Flowcharts, Journal of Computing Sciences in Colleges, *Consortium for Computing Sciences in Colleges*, University of Central Missouri, USA pp: 52 – 60.

[8] Westphal B, Harris F and Fadali M, 2003, Graphical Programming: A Vehicle for Teaching Computer Problem Solving, *33rd ASEE/IEEE Frontiers in Education Conference*, IEEE, Boulder, Colorado, and pp: 19-23.

[9] http://www.cocodex.com/

[10] http://www.Imagix.com/

[11] Danial Hooshyar , Ma.en T. Alrashdan , Masih Mikhak : Flowchart-based Programming Environments Aimed at Novices. ISSN: 2232-1942 Vol. 13 No. 1 January – March 2013.

[12] Rashmi Yadav, Ravindra Patel, Abhay Kothari: Reverse Engineering Tool Based on Unified Mapping Method (RETUM): Class Diagram Visualizations, Journal of Computer and Communications, Oct-2014, 2, 39-49.

[13] Reniers, D., Voinea, L., Ersoy, O. And Telea, A. (2014) The Solid Toolset for Software Visual Analytics of Program Structure and Metrics Comprehension: From Research Prototype to Product. *Science of Computer Programming*, **79**, 224-240. http://dx.doi.org/10.1016/j.scico.2012.05.002.

[14] Auber, D., Melancon, G., Munzner, T. And Weiskopf, D. (2010) SolidSX: A Visual Analysis Tool for Software Maintenance.Poster Abstracts at Eurographics/ IEEE-VGTC Symposium on Visualization.

[15] Aivosto, (2003). Visustin Flow Chart Generator [online]. Aivosto.com. Available from: http://www.aivosto.com/visustin.html [Accessed 21-5-2005].

[16] Kochaporn Suntiparakoo and Yachai Limpiyakorn, Department of Computer Engineering: Recovering Intent of Code from RPG Legacy Source, International Journal of Software Engineering and Its Applications Vol.8, No.3 (2014),spp.291304.s
http://dx.doi.org/10.14257/ijseia.2014.8.3.27.

[17] GREYLING, J., CILLERS, C. & CALITZ, A., (2006), B# The Development and Assessment of an Iconic Programming Tool for Novice Programmers, 7th International Conference on Information Technology Based Higher Education and Training, Ultimo - NSW - Australia, IEEE, pp 376-375.

[18] CARLISLE, M., WILSON, T., HUMPHRIES, J. & HADFIELD, S., (2005), RAPTOR: A Visual Programming Environment for Teaching Algorithmic Problem Solving, ACM SIGCSE Bulletin, 37,1, ACM, New York - NY - USA, pp 176-180.

AUTHOR'S PROFILE

Nivedita Tiwari has received his Bachelor of Engineering degree in Information Technology from R I T Engineering College, Indore in the year 2008. At present she is pursuing M.Tech. With the specialization of Computer Science Engineering in TCET Engineering College. Her area of interest Software Engineering, Reverse Engineering, Data Base Management System.

Lalji Prasad has received his Engineering in Computer Science Engineering. At present he is working as a Professor at TCET College, Indore. His areas of interests are Software Engineering, Reverse Engineering.