Open Source Software: A Study of Dynamic Variance of Complexity

Kanwaljit Singh,
Department of Computer Application.
ACET. Amritsar, India.

Hardeep Singh,
Department of Comp. Sc. & Engg.
GNDU. Amritsar, India.

***Abstract:* Quality and complexity are closely related in software evolution cycle. Evolution period measures the qualities of the software. Software metrics monitor and manage the quality of software. Oscillation in the complexity reflects disparity in the quality. Complexity is one of the indicators of the software quality. Complexity depends upon the size of class, number of statements used and type of statement used in the software development. Video LAN Client (VLC) media player open source software with its 58 versions and 7-Zip open source compression software with 61 versions are used for the quality analysis. This paper measures the complexity of the open source softwares VLC and 7-Zip. The studies of the software complexity are done with comparative analysis on various factors generated by metric tool SourceMonitor. The overview is to calculate the active variation in complexity when compared with functions, class-size, and statements of VLC with 7-ZIP during evolution cycle.**

Keywords: Open source software, structured complexity, metrics, evolution, SourceMonitor, VLC, 7-Zip.

I. INTRODUCTION

Open source software (OSS) is freely available with its source code for study, research, download, modify, and share information [14]. Desirable modules are downloaded quickly from code library and adjusted in the source code of the software to improve its quality with less time and at low cost. This paper introduces two OSS; Video LAN Client (VLC) media player and 7-Zip compression software. VLC consists of clients and server to stream videos across the network. VLC is an open source modular design programmed in an object oriented C++. Incomplete, damaged or unfinished videos can be easily run on VLC.

Fifty eight versions of VLC have been designed since evolution period of 2001 to 2013. Software evolution is the process of developing the software and then frequently renovating it for various reasons. In this paper evolution is a process of improvement, inheriting version and reformation. By the year Feb, 2001 VLC-0.2.0 was released with its properties and complexities. The evolution process starts from the time slot of Feb – April 2001. The evolution in appendix-2 defines the enhancement in the software either by altering the code(s) of previous version as per customer requirements and quality satisfaction. The altitude of complexity gets affected with progression in evolution. Software is distinctly substantial if it fulfills the maximum of seven conditions of Lehmann's Law of evolution [2]. The study of all the versions are compared and evaluated with respect to meticulous metric tool. The metric tool will generate the result in two parameters, i.e. source code in the versions v/s complexity in the modules.

7-Zip is an open source file archive, which may be used to compress and encrypt one or more files for various operating systems. 7-Zip is the conqueror of file archiving and compression tools. It sets the standard for both compression ratio and time with its very own 7z compression format. To compress a file, it manages to beat MagicRAR, WinRAR, and WinZip for the best compression ratio, even with its Fastest Compression setting enabled. 7-Zip takes only 25 seconds to compress target files/folders, WinRAR (44 sec), WinZip (51 sec), and MagicRAR (159 sec). Multinational banking, IT, and other organizations use 7-Zip software for compressing and encrypting the software files for data transfer and storage. Appendix-1 defines the enhancement in 7-Zip in its evolution cycle.

II. PROBLEM DEFINITION

VLC and 7-Zip are open source software with various versions. After the development of one version, another version is ready to release with its own complex properties. This process continues in software evolution course. Each version of software develops with an integration of modules and classes. The statements are the tools used to measure the structure complexity of the software. Difficult and copious statements complicate the modules or classes and inflate the complexity. The quality of the OSS is calculated through complexity value. The organization of elements within the software defines complexity.

The time gap between release date of 1st version and the final version of software should not be the enlarged. This reduces stability of the software. With the time period of 9 and 13 years more than 58 and 61 versions of VLC and 7-Zip have been developed [8]. 58 times in VLC and 61 times in 7-Zip numerous changes of the software will vibrate the market demand and quality of the software

III. METHODOLOGY

To read and study the metrics value of the VLC & 7-Zip software, another OSS is executed called SourceMonitor. SourceMonitor is a metric tool that can calculate 14 metrics of java, C#, C++, VB based software with graphic indicator and filtering techniques to analyse the results [12]. The SourceMonitor (SM) scrutinizes how much code software has and identifies the relative complexity of statements and modules in software. VLC and 7-Zip are programmed in C++, and SM runs the software code at high speed, thousands of lines of code per second. SM has friendly graphic user interface (GUI). SM presents the metrics in form of tables or charts, to measure software in several phases of the development process and save the resultant metrics in "checkpoints". SM helps to find out the changes in the software during the cycle of software evolution by using the Lehman Laws of Software Evolution. Table-1 is the result generated by the SourceMonitor with various attributes or metrics used to measure complexity. Selective metrics are used in this paper, as in Appendix 3, 4.

Files	Lines	State-ments	% Bnches	% Cmnts	Class Defs	Methods/ Class	Avg Stmnts/ Method	Max Cmplexty	Max Depth	Avg Depth	Avg Cmplxty	Functions
387	105,153	35,481	19	25.3	37	4.5	8.9	126	9	1.7	4.21	1,396

Table 1: Study of various attributes by SourceMonitor

➢ *Files*: Total number of files measured in the selected package.

➢ *Lines:* Total number of lines in the selected package, without the blank lines at the end of each included file.

➢ *Statements:* Total number of statements in the selected package.

➢ *% Branches*: Statements such as if, else, for, while, goto, break, continue, switch, case, default and return are measured here as a percentage of the total statements.

➢ *% Comments*: Number of total comments divided by total number of lines. Headers and footers, at the beginning and end of files are not taken into consideration.

➢ *Class Size*: Total number of operations and attributes that are encapsulated in method or class.

➢ *Method per Class*: Total number of complex methods in a class.

➢ *Functions*: Total number of functions existing in the selected package.

➢ *Average statements/method*: Total number of statements inside methods in a selected package divided by the number of methods in the package.

➢ *Maximum complexity*: Value of the coupling and cohesion of the most complex function in the selected package.

➢ *Maximum depth*: Maximum nested methods are depth in the selected package. At the beginning of each file its value is zero. It must be pointed out that statements at levels 1 to 8 are recorded, while statements at deeper levels are counted depth 9.

➢ *Average depth*: Depth is total number of methods starting from root method to leaf method in the execution path. Average depth of software is sum of depth of all execution paths divided by total number of execution paths.

➢ *Average complexity*: The average value of all complexity values in the selected package

After measuring the metrics of VLC and 7-Zip software, the paper will perform the comparative analysis of the metrics of two softwares and find that during evolution period which metric(s) follow evolution laws.

IV. RESULT AND ANALYSIS

Various versions of VLC and 7-Zip have been generated during the evolution period. Each version has hundred of packages in it. The individual version of the software is evaluated with SourceMonitor that generates attributes as shown in table-1. The data is generated, collected and analysed through all versions (shown in Appendix 3, 4).

Metric Analysis	**VLC**	**7-Zip**
Statement vs ClassSize	- 0.115 (-ve)	0.870 (+ve)
Avg. Complexity vs Function	0.221 (Low)	0.971 (High)
Statement vs Avg. Complexity	0.142 (Low)	0.970 (High)
Function vs Class Size	- 0.084 (-ve)	0.737 (+ve)
Statement vs Functions	0.745 (+ve)	0.965 (+ve)
Class Size vs Avg Complexity	- 0.068 (-ve)	0.798 (+ve)
Statements vs Method per Class	- 0.260 (-ve)	- 0.382 (-ve)

Table-2: Correlation value of metrics in VLC and 7-Zip

In the study comparative analysis of the metrics of VLC and 7-Zip software has been done. Correlation among metrics is revealed in table-2. Using complexity metric, software team has the capability to indicate problems of software, guide software testing, and estimate software maintenance efforts [15]. Formats designed for comparative analysis in this paper are:

1. *Average complexity vs. Functions*

Structure of the functions and their interrelation that are used to avoid statement redundancy describes the complexity of the software. Relational analysis classifies that the level of average complexity fluctuates with amendment in number of functions. The relationship is explained with the help of data collection for VLC and 7-Zip analysis in fig.1 and fig.2. There is a huge gap between complexity and function metrics values in VLC and 7-Zip. To recover, complexity value is multiplied by an independent constant value to make it relevant to represent two-dimensional graph of complexity and function.

At the initial development in VLC, after the rise in functions at initial stage with fall in functions, there is equivalent fall in average complexity. There is fluctuation in function due to which average complexity descends. Where as in 7-ZIP application there is linear rise in functions and calm complexity at initial level. At the mid stage of evolution, there is strong boost in functions, due to which complexity level increases with low growth rate in VLC application. In 7-ZIP there is low density, high wavy shade enhancement in functions during evolution period at mid stage, due to which there is less frequency change in complexity. At the final stage, with the small rise in number of functions in VLC, there is narrow increment in complexity. In case of 7-ZIP application, at final stage with slight fall in number of functions there is increase in complexity. In table-2 the correlation among average complexity and function is positive in both the software but low in VLC and high in 7-Zip.

Finally there are three parameters found in the relationship among functions and complexity;

I. Sharp Increase in functions – Gentle Rise in complexity.
II. Aslant increase in functions – Tilt Fall in complexity.
III. Syrupy Rise or Fall in functions – Calm complexity.

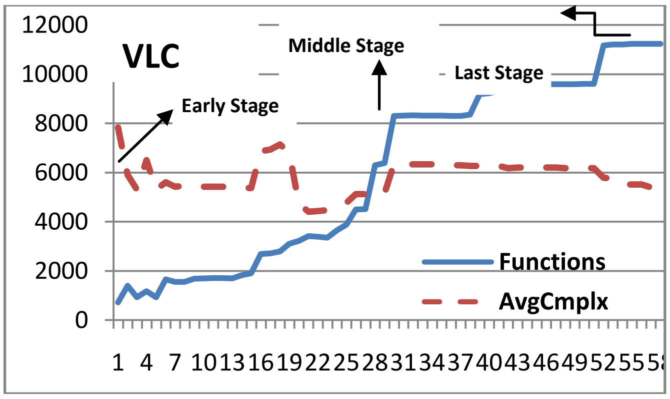

Fig1: Relation among Complexity and Function in VLC

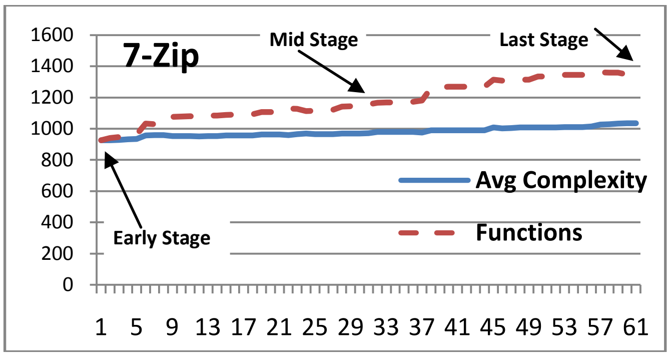

Fig2: Relation among Complexity and Function in 7-Zip

2. *Class Size vs Average Complexity*

Class is the core of the object oriented program, and size of a class has mammoth credence on software output. The status of class structure is measured with the variable called "complexity". Class size is calculated by the multiplication of method/class and statements/method. An analysis is being done on the relation between class size and the complexity in fig. 3. In VLC and 7-Zip, the complexity metric value is multiplied by an independent constant value to make it equivalent to class size for graph development. In VLC, at the early evolution stages, the complexity reduces as the class size remains same. With decrease in class size, average complexity also reduces. In the middle stage of evolution there is a steep fall in class size. With fall in class size, the complexity increases calmly. At the last mode of evolution, there is constant flow of class size. In constant mode of class size, the complexity reduces. This clarifies that the number of statements used in methods at different class remain same but the format of the statements varies.

As compare to VLC in fig.3, 7-Zip application in fig.4 has very narrow variation in complexity with rise in evolution. At the early stage with rise in class size, there is rise is complexity. This results in directs relations. At the mid of 7-Zip evolution period, the complexity increases steadily with bit by bit increase in class size. The complexity gets consistent with change in class size. At the last stage, with minor rise in size, there is narrow increase in complexity. Table2 calculates the correlation among class size and average complexity, it is –ve in VLC and +ve in 7-Zip software. The various parameters found in the relationship among class size and complexity are:

I. Sharp Boost in class size - Minor fall in complexity
II. Fall in class size – Angled increase in complexity
III. Calm in class size – In control complexity

3. *Statement vs Average complexity*

The instruction processed by compiler is called statement. Set of statements is called method. Program is collection of statements, functions and classes. Set of programs develop the software. Statements are the core of the software. Line graph explains the relation between number of statement and complexity level. The complexity metric values of VLC and 7-Zip software are multiplied by different independent constant values to make them equivalent to their corresponding statement metric value.

At the initial evolution development stage of VLC the complexity decreases with increase in number of statements as in fig.5. At middle stage, a sky scraper is generated by number of statements, and a bit increase in complexity.

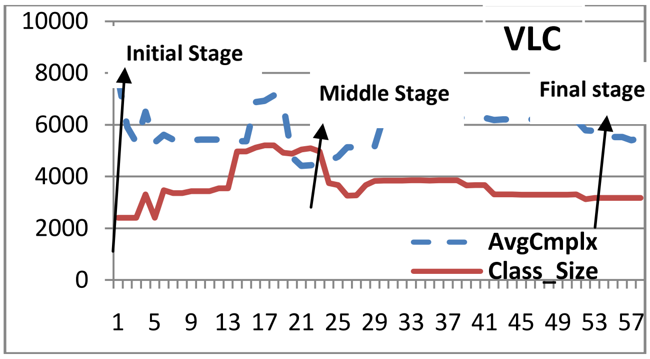

Fig3: Relation among Class-size and Complexity in VLC

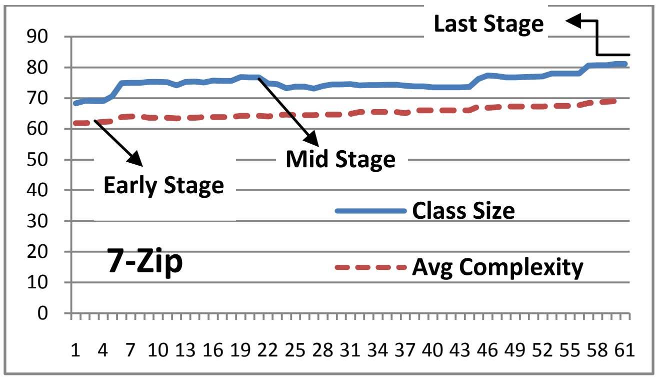

Fig4: Relation among Class-size and Complexity in 7-Zip

At the last level, with narrow rise/fall in statements, there is slim rise/fall in complexity. The less number of complicated statements are modified to large number of simple statements. With increase in statements the complexity level increases.

In the 7-Zip application at fig. 6, there is direct relation among statement and average-complexity at initial level of evolution. With increase in statements, the complexity level also increases. At the middle level of evolution the complexity calmly increase with increase in number of statements. At the final stage of evolution there is instant rise in statements, which makes tiny increase in complexity. The correlation calculation results positive in both the software but low in VLC and high in 7-Zip software in table-2.

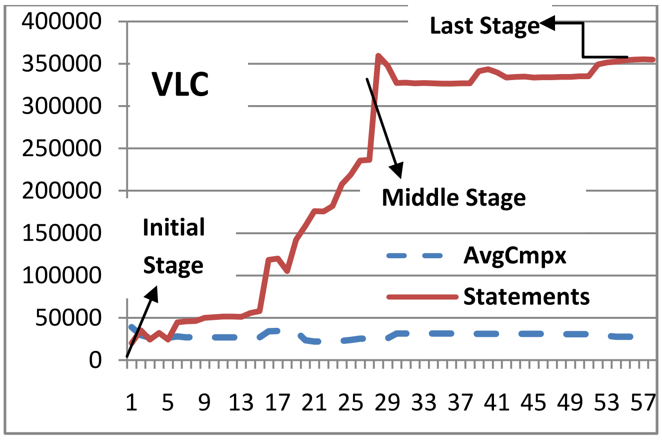

Fig 5: Statement and Complexity relational analysis in VLC

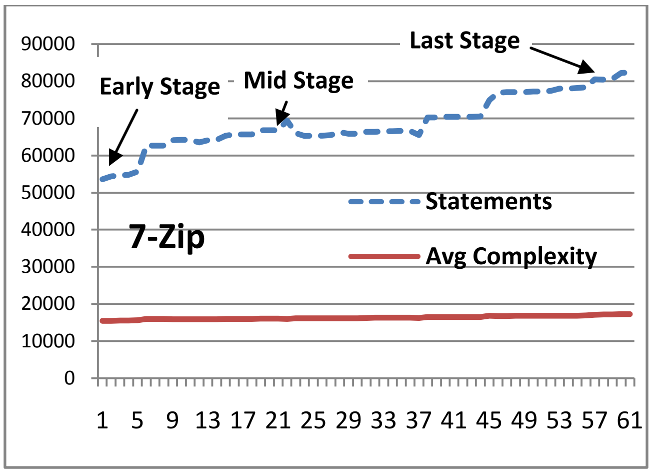

Fig 6: Statement and Complexity relational analysis in 7-Zip

The various parameters found in the relationship are as follow:

1. Strong rise in statements – Potential rise in complexity
2. Steady fall in statements - Slow growth in complexity
3. Slight rise or fall in statements – Constant level of complexity

DISCUSSION

Complexity depends upon the structure of functions. The complexity-function analysis of VLC and 7-Zip in fig-1 & 2, recommends that with increase in evolution, there is increase in functions and the with respect to that the complexity level increases because of coupling and cohesions between functions. Class size is total number of methods and attributes in the structure of the class. In case of VLC it is harder to test, maintain and reuse the class-size during evolution cycle. In case of 7-Zip, it is easy to understand the class-size during the evolution cycle. A team with less number of members will generate fewer errors as compare team with more team members. In statement-complexity analysis of VLC and 7-Zip it is found that at the initial stage because of less number of statements the complexity level was low, as the statements increases, the probability of complexity increases.

CONCLUSION

In this paper we studied the dynamic variability of complexity on evolution of long lived open source programs VLC and 7-Zip. In the study we investigate the implementation of Lehman's Law while evolution to the software. While taking Average Complexity as a major metric and function, statements and class size as minor metrics three various comparative analysis were done between VLC and 7-Zip. On the basis of analysis Lehman's Law is studied. In table-3 we studied that among eight Lehman's law for software evolution at least six are applicable for VLC where as all eight are applicable for 7-Zip.

S. No.	Brief Name	Law	VLC	7-Zip
Law-I	Continuing Change	System continually adapted else they become less satisfactory	Y	Y
Law-II	Increasing Complexity	As an system evolved its complexity increases-unless work is done to maintain or reduce it	N	Y
Law-III	Self Regulation.	System evolution process is self regulating	Y	Y
Law-IV	Observation of Organizational Stability	Global activity rate on a system does not change.	Y	Y
Law-V	Conservation of Familiarity	Developer understand the system behavior. Constant or decline in system growth	Y/N	Y
Law-VI	Continuing Growth	Content of system continually increase to maintain user satisfaction.	Y	Y
Law-VII	Declining Quality	System will decline unless they are rigorously maintained.	Y	Y
Law-VIII	Feedback System	Role of user feedback in providing momentum for future evolution.	Y	Y

Numerically 79% Laws applicable for VLC and 97% are applicable for 7-Zip. This difference of percentage shows that the occurrence of complexity is more in VLC as compare to 7-Zip software.

REFERENCES

1. Jack Zhang, Shikhar Sagar, Emad Shihab, “The Evolution of Mobile Apps: An Exploratory Study”. Proceeding of the 2013 International Workshop on Software Development Lifecycle for Mobile. Pages 1-8. ACM New York, USA, 2013.
2. M.M. Lehman, “ Rules and Tools for Software Evolution Planning and Management”, Journal, Annals of Software Engineering. Vol. 11, Issue 1. November 2001. NJ, USA.
3. Andrea Capiluppi, Maurizio Morisio, Juan F Ramil, “ The Evolution of Source Folder Structure in actively evolved Open Source Systems”. Proceedings of the 10th International Symposium on Software Metrics (METRICS’04). IEEE
4. Rajiv D Banker, Srikant M Datar, “ Software Complexity and Maintainability”. Proceedings of the Tenth International Conference on Information Systems, December 4-6, 1989, Boston, Massachusetts.
5. E Da Wei, “ The Software Complexity Model and Metrics for Object-Oriented” IEEE 2007.
6. Ayman Madi, Oussama Kassem Zein and S. Kadry, “ On the Improvement of Cyclomatic Complexity Metric”. International Journal of Software Engineering and its Applications. Vol.7, No. 2, March 2013.
7. N. Nagappan, B. Murphy, V. R. Basili, “ The Influence of Organizational Structure on Software Quality: An Empirical Case Study”. International Conference on Software Engineering-ICSE, pp: 521-539, May, 2008. Germany.
8. C. Wohlin, B. Lennselius and C. Vrana, "Software Metrics: Structure and Some New Research Results", Proceedings Milcomp, pp. 221-226, London, United Kingdom, 1986.
9. Kanika Raheja, R. Tekchandani,” An Emerging Approach towards Code Clone Detection: Metric Based Approach on Byte Code”. International Journal of Advanced Research in Computer Sc. And Software Engg., Vol. 3, Issue. 5 May 2013. Pg. 881-888
10. Carlos Lopez, E. Manso, Y. Crespo,” The identification of anomalous code measures with conditioned interval metrics”. 13th TOOLS Workshop on Quantitative Approaches in Object-Oriented Software Engineering {(QAOOSE} 2010) – 2010. Malaga, Spain.
11. Gurdev Singh, Dilbag Singh, Vikram Singh, “ A study of Software Metrics”. IJCEM International Journal of Computational Engg. & Management. Vol. 11, January, 2011.
12. Capiluppi A., Ramil J.F. (2004), “Studying the evolution of open source systems at different levels of granularity: two case studies”, Proceeding on the 7th IEEE International Workshop of Principles of Software Evolution, 2004, pp 113 – 118
13. M. Zhang, N. Baddoo, “ Performance Comparison of Software Complexity Metrics in an Open Source Project”. Springer 2007, pg. 160-174
14. Nicholas Drouin, Mourad Badri, “ Investigating the Applcability of the Laws of Software Evolution: A Metrics Based Study” Springer, 2013, Pg: 174-189. Berlin.
15. Nicholas Drouin, Mourad Badri, Fadel Toure, “ Analyzing Software Quality Evolution using Metrics: An Empirical Study on Open Source Software”. Jouranal of Software, Vol. 8 No. 10 Oct. 2013.

Appendix-1 Evolution table of VLC

Sr. No	Version	Date	Sr. No	Version	Date	Sr. No	Version	Date
1	vlc-0.1.99	27 Aug 2000	23	vlc-0.6.1	31 July 2003	45	vlc-1.0.0	06 July 2009
2	vlc-0.2.0	02 Feb 2001	24	vlc-0.6.2	14 Aug 2003	46	vlc-1.0.1	27-july-2009
3	vlc-0.2.60	14 Feb 2001	25	vlc-0.7.0	03 Jan 2004	47	vlc-1.0.2	22 Sept 2009
4	vlc-0.2.70	09 April 2001	26	vlc-0.7.1	02 March 2004	48	vlc-1.1.0	22 June 2010
5	vlc-0.2.70-1	09 April 2001	27	vlc-0.7.2	21 May 2004	49	vlc-1.1.1	21 July 2010
6	vlc-0.2.80	05 June 2001	28	vlc-0.8.0	3 Nov 2004	50	vlc-1.1.2	29 July 2010
7	vlc-0.2.80-1	28 July 2001	29	vlc-0.8.1	14 Nov 2004	51	vlc-1.1.3	18 Aug 2010
8	vlc-0.2.90	10 Oct 2001	30	vlc-0.8.2	25 Jun 2005	52	vlc-1.1.4	27 Aug 2010
9	vlc-0.3.0	09 Oct 2001	31	vlc-0.8.4	26 Nov 2005	53	vlc-1.1.5	13 Nov 2010
10	vlc-0.3.1	06 Dec 2001	32	vlc-0.8.5	6 May 2006	54	vlc-1.1.6	24 Jan 2011
11	vlc-0.4.0	23 May 2002	33	vlc-0.8.6	10 Dec 2006	55	vlc-1.1.7	31 Jan 2011
12	vlc-0.4.1	04 June 2002	34	vlc-0.8.6b	18 April 2007	56	vlc-1.1.8	23 March 2011
13	vlc-0.4.2	10 July 2002	35	vlc-0.8.6c	16 June 2007	57	vlc-1.1.9	12 April 2011
14	vlc-0.4.3	26 July 2002	36	vlc-0.9.0	24 Aug 2008	58	vlc_2.0.0	17 Feb 2012
15	vlc-0.4.4	11 Aug 2002	37	vlc-0.9.1	25 Aug 2008	59	vlc_2.0.1	16 March 2012
16	vlc-0.4.5	14 Oct 2002	38	vlc-0.9.2	14 Sept 2008	60	vlc_2.0.2	27 June 2012
17	vlc-0.4.6	14 Nov 2002	39	vlc-0.9.3	26 Sept 2008	61	vlc_2.0.3	18 July 2012
18	vlc-0.5.0	03 Feb 2003	40	vlc-0.9.4	7 Oct 2008	62	vlc_2.0.4	17 Oct 2012
19	vlc-0.5.1	17 Feb 2003	41	vlc-0.9.5	24 Oct 2008	63	vlc_2.0.5	14 Dec 2012
20	vlc-0.5.2	11 March 2003	42	vlc-0.9.6	5 Nov 2008	64	vlc_2.0.6	7 April 2013
21	vlc-0.5.3	08 April 2003	43	vlc-0.9.8a	3 Dec 2008	65	vlc_2.0.7	26 May 2013
22	vlc-0.6.0	23 June 2003	44	vlc-0.9.9	29 March 2009			

Appendix-2: Evolution table of 7-Zip

Sr No	Version	Date	Sr. No.	Ver-sion	Date	Sr. No.	Version	Date
1	4.13 beta	12/14/2004	22	4.45 beta	4/17/2007	43	9.04 beta	5/30/2009
2	4.14 beta	1/11/2005	23	4.46 beta	5/25/2007	44	9.06 beta	8/17/2009
3	4.15 beta	1/25/2005	24	4.47 beta	5/27/2007	45	9.07 beta	8/27/2009
4	4.16 beta	3/29/2005	25	4.48 beta	6/26/2007	46	9.09 beta	12/12/2009
5	4.17 beta	4/18/2005	26	4.49 beta	7/11/2007	47	9.10 beta	12/22/2009
6	4.18 beta	4/19/2005	27	4.50 beta	7/24/2007	48	9.11 beta	3/15/2010
7	4.19 beta	5/21/2005	28	4.51 beta	7/25/2007	49	9.12 beta	3/24/2010
8	4.2	5/30/2005	29	4.52 beta	8/3/2007	50	9.13 beta	4/15/2010
9	4.23	6/29/2005	30	4.53 beta	8/27/2007	51	9.14 beta	6/4/2010
10	4.24 beta	7/6/2005	31	4.54 beta	9/4/2007	52	9.15 beta	6/20/2010
11	4.25 beta	7/31/2005	32	4.55 beta	9/5/2007	53	9.16 beta	9/8/2010
12	4.26 beta	8/5/2005	33	4.56 beta	10/24/2007	54	9.17 beta	10/4/2010
13	4.27 beta	9/21/2005	34	4.57	12/6/2007	55	9.18 beta	11/2/2010
14	4.28 beta	9/27/2005	35	4.58 beta	5/5/2008	56	9.19 beta	11/11/2010
15	4.29 beta	9/28/2005	36	4.59 beta	8/13/2008	57	9.2	11/18/2010
16	4.30 beta	11/18/2005	37	4.60 beta	8/19/2008	58	9.21 beta	4/11/2011
17	4.31	12/4/2005	38	4.61 beta	11/23/2008	59	9.22 beta	4/18/2011
18	4.32	12/9/2005	39	4.62	12/2/2008	60	9.23 alpha	6/7/2011
19	4.42	5/14/2006	40	4.63	12/31/2008	61	9.25 alpha	9/16/2011
20	4.43 beta	9/15/2006	41	4.64	1/3/2009	62	9.30 alpha	10/26/2012
21	4.44 beta	1/20/2007	42	4.65	2/3/2009	63	9.32 alpha	12/1/2013

Analyisi-3 : 7-Zip Data Analysis

Sr. No.	Statments	Class Defs	Methods /Class	AvgStmts /Method	Class Size	Avg Cmplxity	Functions
1	53649	632	8.32	8.2	68.2	3.09	926
2	54467	636	8.42	8.2	69.0	3.09	940
3	54706	637	8.41	8.2	69.0	3.1	947
4	54832	637	8.41	8.2	68.9	3.11	955
5	55726	649	8.5	8.3	70.6	3.12	963
6	62714	738	8.6	8.7	74.8	3.19	1033
7	62693	737	8.61	8.7	74.9	3.2	1029
8	62699	737	8.61	8.7	74.9	3.2	1029
9	64176	745	8.75	8.6	75.3	3.18	1076
10	64233	745	8.75	8.6	75.3	3.18	1079
11	64243	746	8.74	8.6	75.2	3.18	1080
12	63564	743	8.72	8.5	74.1	3.17	1075
13	64204	746	8.75	8.6	75.3	3.18	1084
14	64251	747	8.76	8.6	75.3	3.18	1084
15	65349	761	8.73	8.6	75.1	3.19	1089
16	65680	763	8.8	8.6	75.7	3.19	1090
17	65757	764	8.79	8.6	75.6	3.19	1093
18	65757	764	8.79	8.6	75.6	3.19	1093
19	66812	772	8.83	8.7	76.8	3.21	1106
20	66807	772	8.82	8.7	76.7	3.21	1108
21	66811	772	8.82	8.7	76.7	3.21	1108
22	69414	816	8.69	8.6	74.7	3.2	1130
23	66034	754	8.66	8.6	74.5	3.22	1128
24	65245	749	8.51	8.6	73.2	3.23	1113
25	65363	747	8.57	8.6	73.7	3.22	1115
26	65365	747	8.57	8.6	73.7	3.22	1115
27	65539	752	8.59	8.5	73.0	3.22	1121
28	66361	763	8.59	8.6	73.9	3.23	1141
29	65919	758	8.65	8.6	74.4	3.23	1145
30	65919	758	8.65	8.6	74.4	3.23	1145
31	66409	763	8.67	8.6	74.6	3.24	1156
32	66368	767	8.52	8.7	74.1	3.27	1166
33	66562	768	8.53	8.7	74.2	3.27	1168
34	66569	768	8.53	8.7	74.2	3.27	1168
35	66637	768	8.54	8.7	74.3	3.27	1172
36	66641	768	8.54	8.7	74.3	3.27	1172
37	65527	756	8.51	8.7	74.0	3.25	1181
38	70277	822	8.29	8.9	73.8	3.3	1262
39	70256	822	8.29	8.9	73.8	3.3	1265
40	70457	824	8.26	8.9	73.5	3.3	1269
41	70460	824	8.26	8.9	73.5	3.3	1269
42	70473	826	8.26	8.9	73.5	3.3	1270
43	70473	826	8.26	8.9	73.5	3.3	1270
44	70503	826	8.27	8.9	73.6	3.3	1269
45	74979	868	8.28	9.2	76.2	3.36	1315
46	76939	891	8.41	9.2	77.4	3.34	1308
47	77087	893	8.39	9.2	77.2	3.35	1310
48	77113	895	8.34	9.2	76.7	3.36	1315
49	77113	895	8.34	9.2	76.7	3.36	1315
50	77233	893	8.35	9.2	76.8	3.36	1336
51	77244	893	8.36	9.2	76.9	3.36	1336
52	77433	893	8.37	9.2	77.0	3.36	1338
53	78057	899	8.39	9.3	78.0	3.37	1346
54	78055	899	8.39	9.3	78.0	3.37	1346
55	78202	900	8.39	9.3	78.0	3.37	1346
56	78396	893	8.39	9.3	78.0	3.38	1350
57	80507	905	8.48	9.5	80.6	3.42	1362
58	80477	903	8.49	9.5	80.7	3.43	1359
59	80621	903	8.49	9.5	80.7	3.44	1360
60	82282	935	8.45	9.6	81.1	3.45	1347
61	82348	936	8.45	9.6	81.1	3.45	1347

Analysis-4 VLC Data Analysis

Sr No .	Statemnts	Class Defs	Methods /Class	Avg Stmts /Method	Class Size	Avg Cmplxty	Functions
1	20407	16	5	9	40	3	720
2	35481	37	5	9	40	2	1396
3	24440	19	5	9	40	2	935
4	32677	44	8	7	55	2	1182
5	24440	19	5	9	40	2	935
6	44854	80	8	7	58	2	1658
7	45952	57	9	7	56	2	1560
8	46328	57	9	7	56	2	1558
9	50358	66	9	7	57	2	1687
10	50921	66	9	7	57	2	1696
11	51662	66	9	7	57	2	1710
12	51905	67	9	6	59	2	1710
13	51503	67	9	6	59	2	1700
14	55894	76	11	8	83	2	1823
15	58053	77	11	8	83	2	1909
16	118561	253	10	9	85	2	2690
17	120155	258	10	9	87	2	2716
18	105325	266	10	9	87	2	2791
19	142414	329	9	9	82	2	3106
20	158881	409	9	9	82	2	3220
21	176169	402	9	9	84	1	3418
22	175863	407	9	9	85	1	3389
23	182272	488	9	9	83	1	3355
24	207721	635	8	8	62	2	3647
25	219210	695	8	8	61	2	3886
26	235789	694	7	8	54	2	4498
27	236551	697	7	8	54	2	4516
28	359436	1337	7	8	61	2	6301
29	347779	1382	8	8	64	2	6385
30	327316	1597	8	8	64	2	8304
31	327696	1596	8	8	64	2	8320
32	327118	1596	8	8	64	2	8333
33	327294	1594	8	8	64	2	8324
34	326946	1594	8	8	64	2	8324
35	326572	1595	8	8	64	2	8325
36	326617	1595	8	8	64	2	8309
37	327033	1595	8	8	64	2	8311
38	326899	1595	8	8	64	2	8359
39	341363	1693	7	8	61	2	9200
40	343463	1707	7	8	61	2	9228
41	339849	1711	7	8	61	2	9337
42	333769	2047	7	8	55	2	9581
43	334595	2047	7	8	55	2	9603
44	334881	2048	7	8	55	2	9603
45	333793	2048	7	8	55	2	9593
46	333935	2048	7	8	55	2	9593
47	334029	2048	7	8	55	2	9594
48	334495	2048	7	8	55	2	9594
49	334419	2048	7	8	55	2	9592
50	335196	2051	7	8	55	2	9607
51	335229	2051	7	8	55	2	9617
52	349089	2118	7	8	52	2	11170
53	351390	2119	7	8	53	2	11209
54	352553	2119	7	8	53	2	11211
55	354039	2120	7	8	53	2	11236
56	354739	2120	7	8	53	2	11236
57	355094	2123	7	8	53	2	11241
58	354995	2123	7	8	53	2	11241