Proc. of the Intl. Conf. on Advances In Engineering And Technology - ICAET-2014

ISBN: 978-1-63248-028-6 doi: 10.15224/ 978-1-63248-028-6-01-12

Predictive Object Point Metrics (POP): A better Size Estimator for OO Software

Shubha Jain[1] and Prof. Raghuraj Singh[2]

***Abstract*— Various Techniques exists for OO estimation during different phases of software development process. PRICE Systems has developed a metric called Predictive Object Points which was designed specifically for Object oriented software and results from the measurement of the object-oriented properties of the system. Another well established technique is function points measurement for size estimation. In order to determine the suitable metrics for OO software estimation, Predictive Object Point (POP) software sizing metric is compared here with well established Function Point (FP) software sizing metric. Various projects have been taken for this comparison. This paper presents the results for these metrics. Both the results are compared to show that POP count estimate is better than FP with certain conditions.**

Keywords*— Object-Oriented Measurement, Predictive Object Point, POP Count, Software Metrics, Software Sizing, Software Estimation Technique, Function Point.

I. Introduction

Software sizing is a process which is used to estimate the size of software. This is an important factor that affects the cost and time of the software project [3, 4, 5, 6, 7]. Predictive Object Points (POP) [2] is a sizing metric which is considered to be the better indicator of size of object oriented software than any other sizing measures like Function Points [8, 9]. POPs are a metric suitable for estimating the size of object oriented software however there is no real mapping of POP with software size exists.

The practitioners may use POP metric to estimate the effort required to complete the project by using the COCOMO II model [3, 10]. As the model uses Kilo Source lines of code for effort estimation therefore a technique was required to convert POP to Source Lines of Code. This has been proposed to be done using simple linear regression analysis [1].

[1]Shubha Jain
Kanpur Institute of Technology, Kanpur
India

[2]Prof. Raghuraj Singh
Harcourt Butler Technological Institute, Kanpur,
India

II. POP Metric Overview

POP was introduced by Mickiewicz in 1998. PRICE systems [2] has developed the POP metric for predicting effort required for developing an object oriented software system. It was designed specifically from results on measurement of the object-oriented properties for Object oriented software systems. It fulfilled almost all the criteria of OO concepts and was based on the counting scheme of function point (FP) method as used in function/procedure oriented software development environment. POPs are intended as an improvement over FPs by drawing on well-known metrics associated with an object oriented system. POPs are suitable metrics for estimating the size and subsequently the effort required for development of object oriented software, based on the behaviors that each class is delivering along with top level inputs describing the structure of a system.

III. Mapping Pop Metric with Software Size

A mapping was suggested between KLOC and POP by using simple linear regression equation [12]. The method gave more accurate results when more and more number of the project is added up in the process.

It is also found that Predictive Object Point Metrics can be used to estimate the size of the Object Oriented software Systems through regression method only for the projects which are developed in the same environment and are built for the same application [13]. However POP may not be related to the size through the same formulation for different types of projects built for different applications. Hence again no direct relation between POP and Software size is found in terms of Source Lines of Code.

IV. Description of Empirical Study

For comparison of POP count results with FP results, three projects [11] have been chosen. The SLOC and POP values are obtained through an APA tool [9]. Table I shows the projects with SLOC, TLC and POP Count values measured through the Tool.

Proc. of the Intl. Conf. on Advances In Engineering And Technology - ICAET-2014

ISBN: 978-1-63248-028-6 doi: 10.15224/ 978-1-63248-028-6-01-12

TABLE I. DETAILS OF PROJECTS ANALYZED

Project No.	Project Name	No. of Java Files	SLOC	TLC
1.	ATM_Banking_Sys	12	1186	6
2.	JaimBot_Ver_1.4	30	4413	33
3.	JaimLib_Ver_0.4	45	1505	44

The size of the project can be estimated in thousands of delivered source instructions (KLOC) and then uses a non-linear equation to determine the effort for projects [14][15]. The formula is given by equation (1).

$$\text{Effort} = a*(\text{KLOC})^{b}\ \text{PM} \qquad (1)$$

The parameters 'a' and 'b' are those that hold the interest. The idea is to define them based on the characteristics of project. These characteristics are then compared with historical data to yield the right numbers for these parameters. This model is really simple. Boehm's [1981] considered three modes of software development in this model: organic, semi-detached and embedded, these are discussed in Table II.

TABLE II. THE COMPARISON OF THREE COCOMO MODES [10]

Mode	a	b	Project size	Nature of Project
Organic	2.4	1.05	Typically 2-50 KLOC	Small size project, experienced developers in the familiar environment. e.g., Payroll, Inventory projects etc.
Semidetached	3.0	1.12	Typically 50-300 KLOC	Medium size project, Medium size team, Average previous experience on similar projects. e.g., database systems
Embedded	3.6	1.20	Typically over 300 KLOC	Large projects, Real time systems, Complex Interfaces, Very little previous experience. e.g., ATMs systems.

The effort for each project is calculated, based on the characteristic of project being considered as shown in Table III.

TABLE III. METRIC, POP AND EFFORT VALUES FOR CHOSEN PROJECTS

Project No.	Project Name	SLOC	POP Count	Effort Calculated
1.	ATM_Banking_Sys	1186	82.9576	2.7146
2.	JaimBot_Ver_1.4	4413	674.1781	9.78
3.	JaimLib_Ver_0.4	1505	581.582	3.104

On Comparing the POP count for any two projects say P_1 and P_2, we find how much times the project P_1 is to project P_2 in terms of size.

This POP Count ratio has been evaluated for different combinations of projects. This ratio is compared with the effort ratio for the same projects. Fig. 1.1 shows the sample POP Count value through APA Tool.

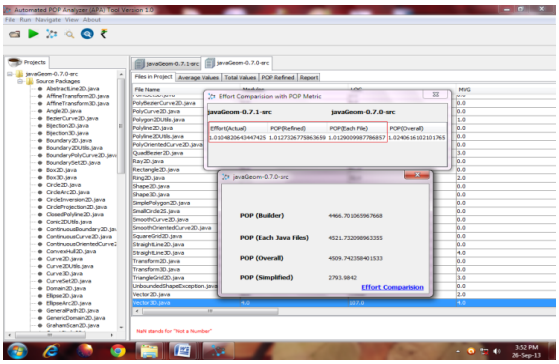

Fig.1.1 Sample POP Count Value

The closer this ratio is to that of the efforts ratio, the more accurate the POP technique is. This is because effort and POP count are proportional, where the constant of proportionality is the POP productivity rate [8]. These results are summarized in Table IV.

TABLE IV. COMPUTED EFFORT, POP AND SLOC RATIOS FOR CHOSEN PROJECTS

S.No	Projects Compared	SLOC Ratio	POP Count Ratio	Effort Ratio
1.	P1/P3	0.7880	0.1426	0.8745
2.	P2| P3	0.4589	0.4355	0.4541

While comparing the projects P1 and P3, POP ratio of both the project is found to be 0.1426 however the actual effort ratio is 0.8745. When this is compared with FP ratio, based on the lines of code (SLOC) metric is found to be 0.7880, which seems to be closer to the effort ratio. However the POP ratio is far apart from effort ratio.

On the other hand if we compare the projects P2 and P3, POP ratio of both the project is found to be 0.4355 however the actual effort ratio is 0.4541. When this is compared with FP ratio, based on the lines of code (SLOC) metric is found to be 0.4589, which seems to be quite close to effort ratio as well as POP ratio.

v. Analysis and Results

POPs rely on a corresponding metric for the Productivity Rate of the organization. The effort then can then be calculated for a particular development by the formula below using the POP count:

ISBN: 978-1-63248-028-6 doi: 10.15224/ 978-1-63248-028-6-01-12

Effort = (Number of POPs) / (POP Productivity Rate)

The POP ratio for projects is actually smaller than the effort ratio of corresponding projects and an analysis shows that the true ratio is probably smaller still:

$$POP_A = \text{Productivity } Rate_A * Effort_A$$

$$POP_B = \text{Productivity } Rate_B * Effort_B$$

$$\frac{POP_A}{POP_B} = k * \frac{Effort_A}{Effort_B} < \frac{Effort_A}{Effort_B}$$

TABLE V. CONSTANT K VALUES FOR CHOSEN PROJECTS

S.No	Projects Compared	SLOC Ratio	POP Count Ratio	Effort Ratio	K
1.	P1/P3	0.7880	0.1426	0.8745	0.18
2.	P2| P3	0.4589	0.4355	0.4541	0.95

So in order for the POP estimate to be better than the FP calculation, k has to be greater than 0.41, which is the POP ratio divided by the FP ratio or SLOC ratio. The POP count can then considered to be more accurate in this case.

In the above study, it may be observed that the POP metric, as an indicator of software size give more accurate results as compared to the FP metric when applied to two of the projects provided that constant k has to be greater than 0.41 .

VI. Conclusion

In conclusion, the POP metric when applied to projects gave a better indication of their size than did the FP metric. This is very important as usually developers as well as researchers express the size of an object-oriented project in function points for the purposes of estimation. However Function Point was not originally intended with object-oriented systems. Here POP metric might be best applied to the estimation of the Object Oriented Systems.
However, still more number of projects under various categories, may be taken for analysis in order to ensure the validity.

References

[1]. Online Statistics Education: An Interactive Multimedia Course of Study, Developed by Rice University (Lead Developer), University of Houston Clear Lake, and Tufts University URL: http://onlinestatbook.com/2/regression/intro.html

[2]. Arlene F. Minkiewicz, "Measuring Object Oriented Software with Predictive Object Points" PRICE Systems, L.L.C. 609-866-6576, 1997.

[3]. B. W. Boehm, Estimating Software Costs, Prentice Hall, N.J., 1981,

[4]. T. C. Jones, Estimating Software Costs, McGraw-Hill, New York, 1998.

[5]. J. Albrecht and J.E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation", IEEE Transactions on Software Engineering, vol. 9,no. 6, pp. 639–648, 1983.

[6]. F. Kemerer, "An Empirical Validation of Software Cost Estimation Models", Communications of the ACM, vol.30, no. 5, pp. 416-429, 1987.

[7]. Kitchenham, "Using Function Points for Software Cost Estimation-Some Empirical Results", 10thAnnual Conference of Software Metrics and Quality Assurance in Industry, Amsterdam, the Netherlands, 1993.

[8]. T. R. Judge, A. Williams ," OO Estimation – an Investigation Of The Predictive Object Points (POP) Sizing Metric in an Industrial Setting", Parallax Solutions Ltd, Coventry, UK.

[9]. Shubha Jain, Vijay Yadav and Prof. Raghuraj Singh, (2013). "OO Estimation Through Automation of Predictive Objective Points Sizing Metric", International Journal Of Computer Engineering and Technology (IJCET) Volume 4, Issue 3, May- June (2013), pp. 410-418.

[10]. Center for Software Engineering, (2000) "COCOMO II Model Definition Manual," Computer Science Department, University of Southern California, Los Angeles, Ca.90089, Available At:http://csse.usc.edu/csse/research/COCOMOII/cocomo_main.html

[11]. http://sourceforge.net/ projects / java-game-lib/files/? Source = navbar

[12]. Shubha Jain, Vijay Yadav, Raghuraj Singh, "An Approach for OO Software Size Estimation using Predictive Object Point Metrics", INDIACom-2014; 8th INDIACom- 2014; International Conference on "Computing for Sustainable Global Development. Paper ID: 57, ISSN 0973-7529 and ISBN 978-93-80544-10-6 serials, IEEE Conference ID 32558, IEEE sponsors: Delhi Section, Other sponsor: Bharati Vidyapeeth Institute of Computer Applications and Management, New Delhi, India.

[13]. Shubha Jain, Vijay Yadav, Prof. Raghuraj Singh, "Mapping Predictive Object Points(POP) with Software Size in OO Environment", International Journal of Enhanced Research in Science, Technology & Engineering, Paper Id: IJERSTE-140198A., ISSN No: 2319-7463., Impact Factor : 1.252, in the current issue, Vol. 3, Issue 1, January-2014.

[14]. I. Sommerville, (2001), "Software Engineering", Sixth Edition. Addison- Wesley Publishers Limited, 2001.

[15]. K. K. Aggarwal and Yogesh Singh., (2006), "Software Engineering", Revised 2nd Edition, New Age International (P) Limited, Publishers, New Delhi.

About Author (s):

Shubha Jain is BE (Electronics and Communication) and M.Tech. (CS). She is pursuing PhD in Computer Science from Uttrakhand Technical University, Dehradun in area of Software Engineering. She has about 20 years of experience in teaching. Currently she is working as Associate Professor and Head in CSE/IT Dept at Kanpur Institute of Technology, Kanpur. She is the Secretary, Kanpur Chapter, CSI and member of IETE society. She has guided 5 M.Tech. Projects and several B.Tech projects. She has 10 papers in National/International Conferences/Journals.

ISBN: 978-1-63248-028-6 doi: 10.15224/ 978-1-63248-028-6-01-12

Prof. Raghuraj Singh is B.Tech. (CSE), M.S. (Software Systems) and Ph.D. in Computer Science and Engineering from U.P. Technical University. He has about 23 years of experience in teaching. Currently he is working as Professor and Head in the Department of Computer Science & Engg. at HBTI, Kanpur. He has guided 7 PhDs and 17 M.Techs and several B.E./B.Tech projects. He is the Chairman, Kanpur Chapter, CSI, Life Member of ISTE, Member of the Institution of Engineers (India), Fellow Member of IETE, Professional member of ACM and Senior Member of International Association of IACSIT. He has more than 80 papers in National / International Conferences and Journals to his credit. Currently 4 students are working for PhD and 4 are pursuing M.Tech. under his guidance.