International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-11, September 2019

OOFFMM: A Size Estimation Model for Real Time Application

Sheeba Praveen, Devendra Agarwal

Abstract: ***A lot of Size Estimation technique has been previously proposed for estimation of the effort of object oriented software projects. After reviewing several object oriented metrics we found that FFP is most excellent right metrics for real occasion application. We took FFP metric for estimating the size. There control functions will helps to calculate those complex, scientific or real time process that were not easily estimated by FP or any other metrics. This research job mostly focus on formative the functional_size of Real-time application at early stage. We define the rules to identify logical files from UML diagrams on different models of OMT. This proposed model is completely base on OOD Methodology owing to the version of the Function_Point Analysis to Object_Point Analysis .We also presented the result obtained by applying the model in a Case study. The result has proven that discovered OOFFA metric and Effort estimation model (OOFFMM) is the best model and metric suite for object oriented MIS applications as well as real time applications.***

Keywords: OOFFMM, OOFFP, OOILF, OOEIF, OOSC, OOAG, OOMX, OOAS, OOGEN, Real Time, Control function Count. OO Metric suite.

I. INTRODUCTION

Now days improved management of software projects demands improved methods for measure the process. For this reason we require best measurement model for improvement of product quality from the early stages of software development. According to our hypothesis that if we calculate defect by using best error-prone metric then we can add to the excellence of software. Subsequent to theoretical as well as practical studies we originate that OOFFMM metric suit be most excellent fault flat metric. The three diverse quality metric suites are [1]. Chidamber plus Kemerer (CK) Metrics produces statistical models that is effective in detect error-prone in classes[2]. Robert C. Martin's Metric Suite is high-quality in predicting the fault in conditions of the packages[3]. McCabe's Metric Suite are high-quality fault-proneness predictors in method .McCabe's Cyclomatic Number of OOFFP is most excellent metric to events errors in the code[4]. One of the primary reasons to take OO approaches for the reason that it joystick difficulty of a system by supporting hierarchical disintegration all the way by both data as well as step by step thought. The rest of the paper is prearranged as follows section 2describe the future size estimation (OOFFMM) model. Section 3describe the OOFFP counting procedures and a case study and how to use the counting procedures on it, section 4 describes result calculations and analysis, section 5 describes the critical findings and last section is conclusion.

Manuscript published on 30 September 2019.
*Correspondence Author(s)
Sheeba Praveen is an Assistant Professor in Computer Science & Engineering Department ,Inegral University, Lucknow, U.P.
Agarwal, Devendra is currently working as Prof. & Head at Department of Computer Science,School of Engineering, BBD University, Lucknow. U.P.

II. PROPOSED OOFFMM SIZE ESTIMATION MODEL AND METRIC SUITE

In this paper we are proposing Object Oriented Full Function Metric Model (OOFFMM) which is extraordinarily helpful for evaluating the size just as exertion of all sort of object Oriented MIS and ongoing undertakings.

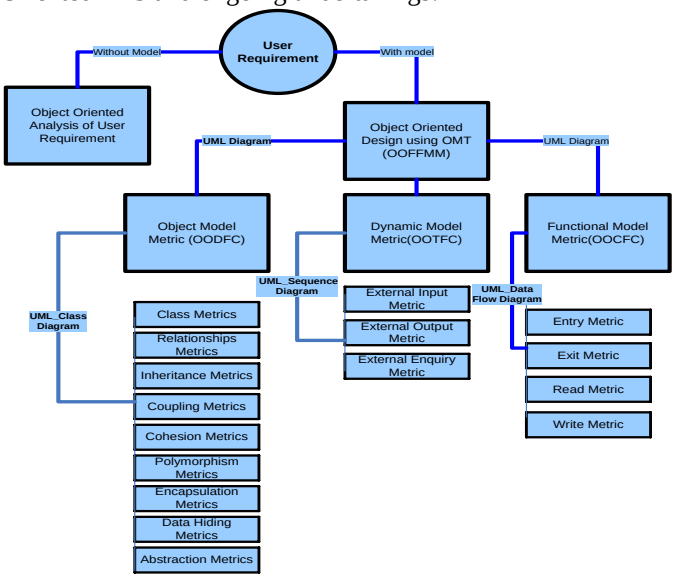


Figure: 1 Size estimation model

A. Proposed OOFFMM Metric suite

OOFFMM metrics Suite were designed based on Object Oriented Methodologies such as the usage of inheritance, polymorphism, Coupling , Cohesion , the amount of responsibilities in a class, etc. to count all the function of Object oriented MIS software as well as Real Time software at early stage. Brief introductions of following metrics are as follows:

OOFFMM Suite		
Class Metric	NPMC	Number of Public Methods Count
	WMC	Weighted methods per class Count
	LOCC	Lines of Code Count
	NOMC	Number of Method count
	AMC	Average Method Complexity
	CCC	The McCabe's Cyclomatic complexity Count
Relationship Metric	SCc	single class count
	Asc	Associated class count
	AGc	Aggregation Class count
	GNc	generalized class count
	MXc	Mixed Class Count

Inheritance Metrics	NOCC	Number of Children Count
	DITC	Depth of Inheritance Tree Count
	IC	Inheritance Coupling Count
Coupling Metrics	CE	Efferent coupling
	CBO	Coupling between object classes
	CA	Afferent couplings
	CBM	Coupling Between Methods
Cohesion Metrics	LCOM	Lack of cohesion in methods
	CAM	Cohesion Among Methods of Class
Polymorphism Metrics	NOP	Number of polymorphic methods
Abstraction Metrics	MFA	Measure of Functional Abstraction
Data hiding Metrics	DAM	Data Access Metric
Transactional Function Count Metrics	EIC	External Input Count
	EOC	External Output Count
	EQC	External inquiry
	RFC	Response for a Class
Control Function Count Metrics	UCGC	Updated Control Group Count
	RCGC	Read-Only Control Group count
	ECEC	External Control Entry count
	ECXC	External Control Exit Count
	ICRC	Internal Control Read Count
	ICWC	Internal Control Write Count

Table1: OOFFMM Suite

III. Object Oriented Full Function Point Counting Process

FFP measurement rules are defined in the IFPUG: Full Function Points counting practice Manual[5]. The main aim is to calculate the Full Functions like ILFs, EIFs, EIs, EOs, EQs, as well as control functions like UCGs, RCGs, ECEs, ECXs, CICRs, CICWs from UML Diagrams of projects. After that each functions will classified into three different levels of complexity (simple, Average and Complex).

There are following calculation steps of OOFFPA applied on class diagrams and sequence diagrams. First Determine the Type of FP Count afterward recognize the Counting Boundary next Count OODFC Types at that time Count OOTFC Types then Count new Control then Count new Control TFC Type then Calculate UFP Count after that settle on value adjustment factor along with most recent compute Adjusted Function point. Values of Step3 and Step4 shows the counts for each function kind are mechanically confidential according to difficulty and then prejudiced. The total for all function types is the unadjusted_point count up. In this research work we apply the step 3,4,5 and 6 on starting from the specification of the system in terms of UML diagrams. Here we are explaining all the counts with the help of case study. Here we are choosing Human Emotion Detection (HED) as a case of real time system which helps to analyze and validate the result.

- **Object Oriented Management Function Count (OOMFC)** There are two types of count perform in management function count they are Object Oriented Function Count and Transactional function Count.
- *Object Oriented Data Function count*

Rules for counting management function type:

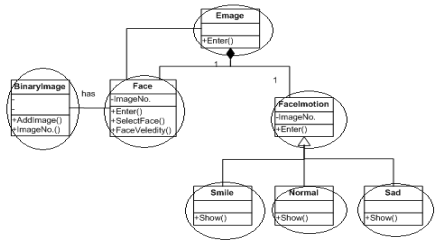
Step1: Select candidates of data functions
Step2: Determine function type
Step3: Judge Complexity of data function
There are two types of count perform in management function count they are

1. Object Oriented Data Function Count (OODFC)
2. Object oriented Transactional function Count (OOTFC)

OODFC counts the Internal and external data necessities call as OOFP. Counting OOFPs is a four footstep procedure:
1. The object model is analysis to make out the units that are to be count as LFs.
2. The difficulty of every LFs. in addition to service ask for is resolute
3. LFs are map to difficulty level of L, A, as well as H.
4. The standards are summed to create the OOFP.

- Classes are normally mapped hooked on logical files; on the other hand relationship between classes (Aggregation and generalization/specialization) can from time to time have need of counting a cluster of classes as a single Logical File.

Dissimilar choices how to contract with aggregation as well as Generalization/ specialization relationships lead to far removed from ways to identify LFs. Here there are four different choices we identify that are



:

Figure: 2SCLFs

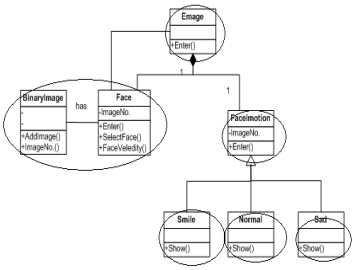


Figure:3AGLFs

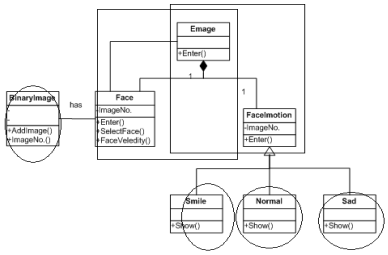


Figure:4ASLFs

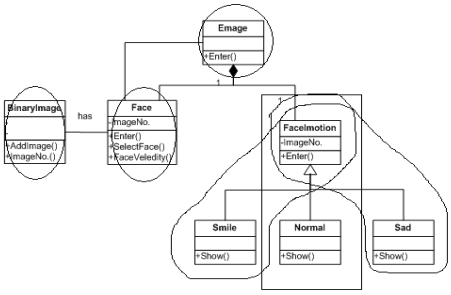


Figure: 5GN/SLFs

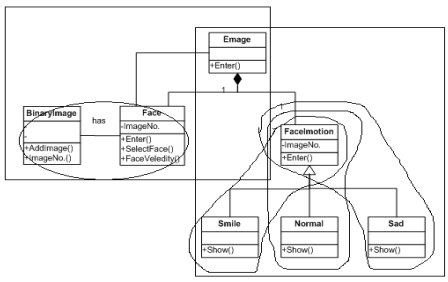


Figure: 6MX LFs

Figures (2)– (6) show four unique ways that classes in an object model may be converged, as per which of the four diverse Logical Files distinguishing proof methodologies is utilized. Here we demonstrate the OOFPs that are figured for every variation. The counting procedure for every individual class gives the DETs and RETs appeared in Figure 12.The class Binary image has Two DETs (one because of the one data items and one due to the numerous to-one association with class Face) and one RET (since the class itself is a gathering of related data items).The class Image has one DET because of its two data items, one RET because of the one-to-numerous accumulation with Face. The Face has one DET and one RET for its very own structure. The class Face Emotion has one DET and Two RETs, one RET because of the one-to-numerous speculation with emotion and one RET for its very own structure. Each other class has one RET for its very own structure and zero DETs as it has no data items. Contingent upon which ILF distinguishing proof tables with every variation. The worth Low is appraised as 7 OOFP, as per the IFPUG tables. The most astounding OOFP check comes when each class is considered a solitary ILF. The various variations have the impact of diminishing the OOFP esteem, as they decrease the quantity of ILFs. In spite of the fact that there is an expansion in DETs/RETs in the consolidated ILFs, it isn't sufficient to raise the ILF complexity to higher values [3].

OOFPs can be calculated as:

$$OOFP = OOFP_{ILF} + OOFP_{EIF} + OOFP_{SR}$$

where:

$$OOFP_{ILF} = \sum_{o \in A} W_{ILF}(DET_o, RET_o)$$

$$OOFP_{EIF} = \sum_{o \notin A} W_{ELF}(DET_o, RET_o)$$

$$OOFP_{SR} = \sum_{o \in A} W_{SR}(DET_o, FTR_o)$$

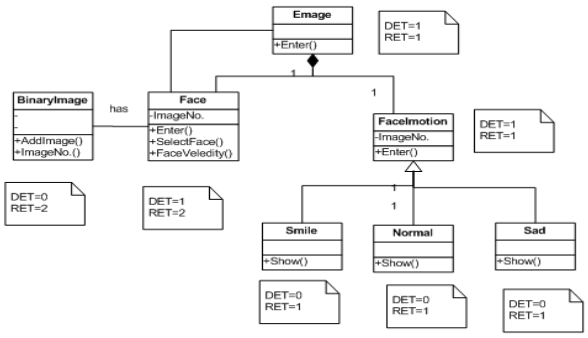


Figure:7 DET & RET calculation for OOILFs

Types	ILFs	SR	OOFP_ILFs	OOFFP_SR	Total_OOFPs
SCc	7	7(L)	49	49+60	109
ASc	6	7(L)	42	42+60	102
AGc	6	7(L)	42	42+60	102
GNc	6	7(L)	42	42+60	102
MXc	4	7(L)	28	28+60	88

Table2:ILF and SR complexity contribution

➢ *Object oriented Transactional function Count (OOTFC)*

Transactional functions stand for the functionality provide to the user for the meting out of data by an submission. Transactional functions are as:

- Object Oriented External input(OOEI)
- Object Oriented External output(OOEO)
- Object Oriented External inquiry(OOEQ)

Rule of including transactional purposetype are:

For every object in the diagrams, we be relevant the subsequent 2 steps to add up the TFC

- Step1: Select candidates of TF: List the series of events that the first msg is send by the actor object as well as the last msg is conventional by the actor object
- Step2: decide the type of transactional function: For everysequenceplanned in Step1, by means of the subsequent 5 different patterns, we settle on the type of the transactional purposealong with the difficulty (DET and FTR) of them .

Patterns of counting OOEI, OOEO, OOEQ:

Pattern 1: An object send msg to a Data Function (see Figure 8). We look upon this prototype as OOEQ. If no msg of view are agreed on the msg, we don't look upon it as OOEI. DET is the number of influence of the msg. FTR is 1 in view of the fact that there is one DF.

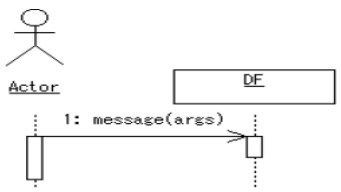


Figure 8: Pattern 1

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Pattern 2: A Data Function send msg to an actor (see Figure 9). In the event that all the perspective of the msg are the indistinguishable as the trait of the DF, we view it as External Inquiry. or there will be consequences, it implies that the msg contain coming about data. from that point forward, we view it as OOEO. DET is the quantity of perspective of the msg. FTR is 1 in perspective on the way that there is one DF.

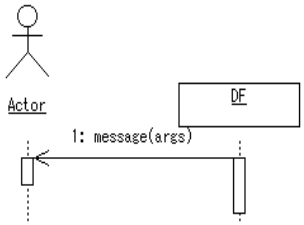


Figure 9: Pattern 2

Pattern 3: An actor item send msg to a DF, besides continues msg on or after the DF to the actor object (See Figure 8and Figure 9). In Figure 16, pay focus to a "msg2", if all the perspective of the "msg2" are equivalent to the trait of the DF, we view two msgs (msg1 and msg2) as one External Inquiry. or something bad might happen, it income that the "msg2" contain basic procedure; we view two msgs as one outside Output. In extra words, we think about that the "msg1" from the actor is the contribution of the contribution for data recovery. DET is the quantity of impact of yielded msg (msg2). FTR is 1 since there is one DF.

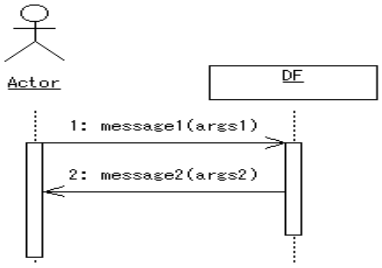


Figure10: Pattern3

Pattern 4: An actor object send msg to a DF notwithstanding the DF sends msg to an extra actor object (see Figure 11) we separate it snared on two Transactional functions. with the expectation of is, Pattern 4 is the blend of the Pattern 1 and the Pattern 2.

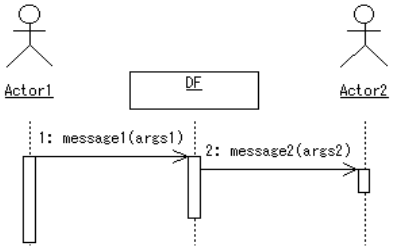


Figure:11 Pattern 4

Pattern 5: An actor object sends msg to a DF too asin end the actor object gets the react msg from side to side in excess of a couple DFs (See Figure12) Pattern 5 handle a succession of msgs. That is the situation that when an actor object send a msg, the appropriate response msg get through a few DFs. For instance, in Figure 12, a grouping of five msgs (msg1, 2, 3, 5, 6) is an applicant of transactional work. right then and there, we apply Pattern3 to msg1 and msg6. In Figure 12, in perspective on the way that msg 4 is excluded in the movement, we be proper other model (for this situation, Pattern2) independently to the msg. DET is the quantity of perspective of yielded msg (msg6). FTR is 2 from when there are two DFs in the grouping.

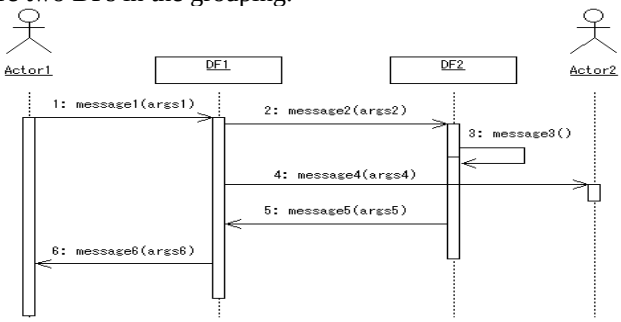


Figure 12: Pattern

The following table shows the functional_complexity for each OOEI:

OOEI	**FTRs**	**DETs**	**Functional Complexity**
Browse_Image	1	4	L
Apply_Skin_colour	1	3	L
Largest_connected_Region	2	10	A
RGB_Image	2	8	A
Binary_Image	1	4	L
Image_Lip	0	2	L
Image_Eye	0	2	L
Image_Eyebrow	0	2	L
Bezier_Curve	1	3	H
OOEOs	**FTRs**	**DETs**	**Functional_ Complexity**
Valid_Face	0	6	H
Connected	2	4	A
Happy_emotion	1	3	L
Fear_Emotion	1	5	L
Surprise_Emotion	1	7	L
Sadness_Emotion	1	4	L
Anger_Emotion	1	6	L
Disgust_Emotion	1	2	L
OOEQs	**FTRs**		**Functional_ Complexity**
Fear_samples	1	2	L
Surprise_samples	1	1	L
Sadness_samples	1	4	L
Anger_samples	1	6	L
Disgust_samples	1	4	L
Happy_samples	1	3	L

Table 3.Functional complexity OOEI, OOEO, OOEQ

- *Object Oriented Control Function Count(OOCFC)*

The Cosmic FFP software model distinguishes four types of data movements, Object Oriented Entry Data Movement, Object Oriented Exit Data Movement, Object Oriented Read Data Movement, Object oriented Write Data Movement

Steps1. Identify The Entry Data Movement from DFD: Entry, data is moved from the user across the process boundary inside the functional process. e.g. Data Input on Button Press (Entry)

Step2. Identify The Exit Data Movement from DFD: Exit, data is moved from inside the functional process across the process boundary to the users. e.g. Data Output on Button Press (Exit)

Step3. Identify the Read Data Movement: Read, moves data inside the process from a persistent data store .e.g. Database. (Read)

Step4. Identify the Write Data Movement: Write, moves data from inside the process to a persistent data store. E.g. Database. (Write)

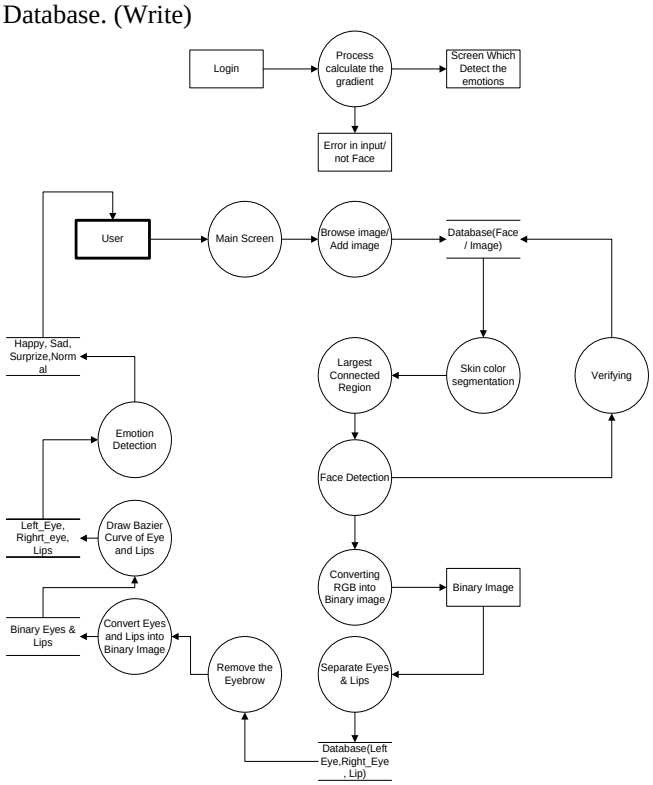


Figure13: UML_DFD of HED

New control data functions types

- **Object Oriented Updated Control Group (OOUCG)**

A UCG is a collection of control data rationalized by the application being counted. It is recognized as of a functional viewpoint. The control data live for more than one transaction.

- **Object Oriented Read-Only Control Group (OORCG)**

An RCG is a group of be in charge of data used, but not rationalized by the submission being count. It is identified on or after a functional perspective. The manage data live for more than one transaction.

New control transactional function types :

- **Object Oriented External Control Entry (OOECE)**

An ECE is a only one of its kind sub-process acknowledged from a functional perspective. An ECE processes be in charge of data approaching from exterior the application's boundary. Example: 1 image will enter for emotion discovery in the submission (control data cross the application boundary). from the time when there is one sub-process desirable to browse icon, consequently 1 ECEs.

- **Object Oriented External Control Exit (OOECX)**

An ECX is a only one of its kind sub-process recognized on or after a functional point of view. An ECX processes manage data leaving outside the application border line. Example: the sub-process that will demonstrate the emotion of human (control data sent outer surface the application boundary) is an ECX. in view of the fact that there is one sub-process desirable to show emotion, consequently 1 ECX.

- **Object oriented Internal Control Read (OOICR)**

An ICR is a only one of its kind sub-process recognized from a functional point of view. An ICR reads manage data. case in point: 7single sub-processes need to read dissimilar human emotions sample (example: happy, sad, normal etc.)at dissimilar times for judgment purposes. as a result, there are 7 ICRs.

- **Object oriented Internal Control Write (OOICW)**

An is a sole sub-process documented from a deliberate stance. An ICW composes oversee information. Representation: 15 special sub-forms call for to apply different methodology to wind up mindful of countenances and feelings of human (model: RGB to Binary change, be proper Bezier curve on face etc.) at diverse times. Consequently, there are 15 ICWs.

Process	Sub_Process Discription	Sub_process
Human_Emotion_Detection	Browse_Image	1
	Face_Detection	15
	Emotion_ Detection	7
	Show_Emotion	1
	Sum	24

Figure 4:Sub-Processes of HED

OOCFC new function types	Description	No. of sub-processes
OOUCG	Data updated by the application	8
OORCG	Data not updated by the application	10
OOECE	Incoming external message	1
OOECX	Outgoing external message	1
OOICR	Referred attribute in an elementary action	7
OOICW	Update attribute in an elementary action	15
	Total	42

Figure 5: OOCFC of HED

IV. RESULT ANALYSIS

A. *Object Oriented ILF and EIF Complexity and Contribution (OOILF):*

The following table shows the total contribution for the OOILF function type.

Type	ILFs	SR	ILF_OOFPs	SR_OOFPs	Total_OOFP

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

SCc	20	7 (L)	20*7	140+60	200
ASc	14	7 (L)	14*7	98+60	158
AGc	14	7 (L)	14*7	98+60	158
GNc	7	10 (A)	7*10	70+60	130
MXc	7	10 (A)	7*10	70+60	130
Type	EIFs	SRs	ILF_OOFPs	SR_OOFPs	Total_OOFPs
SCc	8	5 (L)	8*5	40+60	100
ASc	7	5 (L)	7*5	35+60	95
AGc	7	5 (L)	7*5	35+60	95
GNc	5	10 (A)	5*10	50+60	110
MXc	4	10 (H)	4*10	40+60	100

Table6. Total complexity of OOILF& OOEIF

B. *Object Oriented EI, EO as well as EQ difficulty and Contribution (OOILF)*

The next table shows the total contribution for the OOEI, OOEO and OOEQ function type:

Table7. Total complexity of OOEI, OOEO & OOEQ

Function _Type	Total _Function	L	A	H	Total _Complexity	Total _function type
OEI	9	6*3	2*4	1*6	18+8+6	32
OEO	8	6*4	1*5	1*7	24+5+7	26
OEQ	6	6*3	0*4	0*6	18+0+0	18
Total						76

C. *Object Oriented new Control Data and Transactional Function Count*

The following table shows the total contribution for the OOCC function type.

Function _Type	Total _Function	Low	Average	High	Total _function type
OOUCG	8	6*2	2*3	0*5	*18*
OORCG	10	6*2	3*3	1*5	26
OOECE	1	1*3	0*4	0*6	3
OOECX	1	0*3	1*4	0*6	4
OOICR	7	3*3	3*4	1*6	27
OOICW	14	7*3	7*4	0*6	49
Total					**127**

Table8. Total OOCCs Complexity Rate

D. *work out Adjusted Function Point Count (UFP)*

The following table shows the payment of the application functionality to the unadjusted function point count.

Function_Type	Total _functional _Complexity
OODFC	1341.00
OOTFC	76.00
OOCC	42.00
Total	**1459.00**

Table9. Total UFP of OOFFP

E. *Assumptions & Results*

value _djustment_factor (V_A_F)

Calculate fourteen general_system_characteristics (GSC) on a scale from 0-5 to get the degree of influence (DI). Add DI for all Fourteen GSC to generate the total DI.

V_A_F = (TDI * 0.01) + 0.65

For example, the following VAF is intended if there are 3 DI for each of the Fourteen GSC descriptions

V_A_F = ((3*14) * 0.01) + 0.65=1.07

Calculate adjusted _Function _point count(AFP)

A_F_P = (UFP * VAF)

A_F_P = (1459.00 * 1.07)

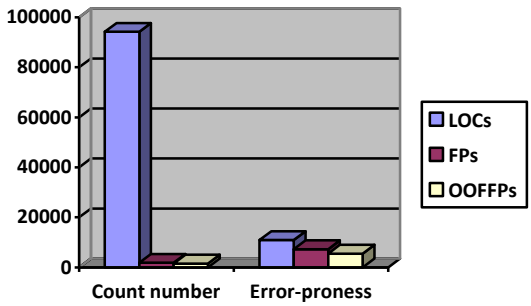
AFP = 1561.13 or 1561.00

As per Previous result analysis shows that one function point convert into 60 times of code because of OOP language so (LOCs = 60 * 1561 = 93660 (approx)). According to researchers most of the software has found an average of 3 errors/FP throughout Software analysis and software design and 4 errors/FP throughout software unit and integration testing. Thus, possible number of errors in analysis and design reviews should be 3*1561 i.e. 4683. At the time of testing Possible number of errors should be 4*1561 i.e. 6244. 00.Thus total possible number of errors should be 10927(approx) [7].

After analysing with different metrics tool it was observed that LOCs are 94181.00, which is more than intended LOCs (on the basis of FPs in initial stage) by a value of 1561.00. At the time of analysis and design reviews the total no of error are 4683.00 and at the time of testing are errors are 6244.00. Thus total errors found are 10927.00 which are more than intended by a value of 1561.00.

Retrieval Number: J93780881019/19©BEIESP
DOI: 10.35940/ijitee.J9378.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)



Metrics	Count number	Error-proness
LOCs	94181	10927
FPs	1850	7259
OOFFPs	1561	5463.5

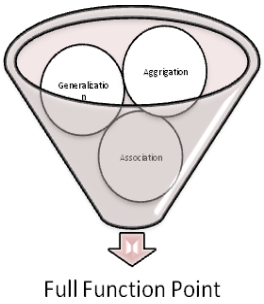
Table 10. Metrics comparison

The above chart demonstrates that the OOFFP measurements is the best for size estimation of programming. By utilizing this size we can figure the Productivity and quality framework. So OOFFMM metric suite improves the exhibition of all sort of MIS just as ongoing programming.

V. CRITICAL FINDINGS

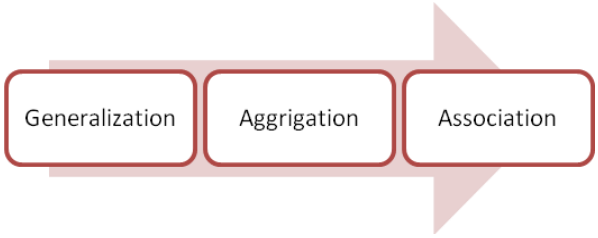
Here in this paper we are not just calculation the Full Function Point. We are also finding these:

- Low-Level designing metrics association with Full function Point.



Here we can address Full function point with the help of Generalization, Aggregation and Association

- Dependency of Full Function Point on Object Oriented Characteristics.



- Show the relationship of these with object-oriented characteristics to match the Full Function point Criteria
- Association +f Full Function Point with Data Flow Model of Object-Oriented Methodology.

After assessing and measuring point 1 and point 2, we will be able to show the dependency and its association with full function point in the functional model, i.e. Data flow diagram

VI. CONCLUSION

The main objective of OOFFMM suite is to count all the functions in all perspective. After literature review we investigated that there are four ISO standard theoretical and conceptual method have been approved by ISO and become an international standard that are COSMIC FFP, IFPUG FPA, Mark II FPA ,NESMA FPA.These methods and metrics were designed to examine mostly length and functionality attributes of software. Our Findings shows that no one has proposed complete metric suite whereby solved the simple functions as well as complex functions related issues. Here we proposed class size metric from static model which will count the data and functions(ILFs, EIFs) associate to low level metrics from UML_class diagram, transactional metric from Dynamic model which will count the functions(FTRs, DETs, RETs) from UML_Sequence Diagram and Control Function metric from Functional model which will count the control functions (Input, output, read write) from UML_DFD. OOFFMM metric suite will be applicable for all projects and able to estimate the size and efforts of complex functions. Thus, the continuous segment study is mainly focus low level observations using with object-oriented characteristics to match the Full Function point Criteria and use the method of multiple linear regression.

REFERENCES

1. N. E. (1996). Software Metrics - A Rigorous & Practical Approach. London: International Thomson Computer Press.
2. M. S. K. (0AD). , Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects. IEEE Transactions on Software Engineering, vol. 29, 297–310.
3. S. O. (1999). A Method For Measuring The Functional Size Of Embedded Software. 3rd International Conference.
4. G. K. (2015). A Study on Robert C.Martin's Metrics for Packet Categorization Using Fuzzy Logic. , International Journal of Hybrid Information Technology, 8.
5. W. L. (1998). Another Metric Suite for Object-Oriented Programming, Journal of Systems and Software, 44(2), 155–162.
6. S. P. (2012). Object oriented Full Function Point Analysis: A Model for Real Time Application. International Journal of Electronics and Computer Science Engineering, 2409–1416.
7. S.P. (2012). Object-Oriented Full Function Point Analysis: An Empirical Validation, International Journal of Computational Engineering Research (ijceronline.com) Vol. 2 Issue. 8.
8. F. (1996). Software Metrics - A Rigorous & Practical Approach. : International Thomson Computer Press.
9. Ç. G. (2006). A Case Study on the Evaluation of COSMIC-FFP and Use Case Points . Published in the Proceedings of Smef.
10. C.-J. L. (2016). A Software Maintenance Project Size Estimation Tool Based On Cosmic Full Function Point. 2016 International Computer Symposium (ICS), 1, 555–560. doi: 10.1109/ICS.2016.0115
11. G. S. (2011). A Study of Software Metrics. IJCEM International Journal of Computational Engineering & Management, 11.
12. E. L. (2014). A System for Measuring Function Points from an ER-DFD Specification. Research Gate.
13. A. (2017). CNMES 2017 Software Cost Estimating with COSMIC - Critical knowledge for today and tomorrow . COSMIC - Common Software Measurement International Consortium.
14. S. K. D. (2012). Comp ar ison Study and Review on Object - Oriented Metrics . Global Journal of Computer Science and Technology, 12(7).
15. A.H. (2018). COSMIC Function Points Evaluation for Software Maintenance. 11th Innovations in Software Engineering Conference.

16. R. D. (2011). COSMIC Function Points: Theory and Advanced Practices . Auerbach Publications.
17. T. (2018). Size and Effort Estimation Based on Problem Domain Measures for Object-Oriented Software. International Journal of Software Engineering and Knowledge Engineering, 28, 219-238.
18. S. D. M. (2016). Web Effort Estimation: Function Point Analysis vs. COSMIC. Information and Software Technology, 72, 90–109.
19. G. (2010). Reuse Metrics for Object-Oriented Method. 2nd International Conference on Information Engineering and Computer Science.
20. Z. Y. (2012). An improved software size estimation method based on object-oriented approach. 2012 IEEE Symposium on Electrical & Electronics Engineering (EEESYM).
21. S. P.(2012)"Systematic Literature Review Of Software Development Project Size Estimation Metric: OOFF" JASC: Journal of Applied Science and Computations.

AUTHORS PROFILE

Sheeba Praveen is an Assistant Professor in Computer Science & Engineering Department ,Inegral University, Lucknow, U.P. She received bachelor and master degree in Computer Sciennce andEngineering from Integral University Lucknow. and persuing P.hd in Object Oriented Metricesfrom Babu Banarsi Das University, Lucknow.She has over 9 years of teaching experience.She haspublished 10 research papers in international/national journals and conferences proceedings. Her research interest are software engineering, Object Oriented, UML etc .Email:er.sheeba.iu@gmail.com

Agarwal, Devendra is currently working as Prof. & Head at Department of Computer Science,School of Engineering, BBD University, Lucknow. He has over 18 years of teaching & 5 years ofindustrial experience. He has done his B.Tech in CS from Mangalore Universityin 1993, M.Tech in Computer Scince from U.P.Technical University, Lucknow in 2006, andPh.D. from Shobhit University, Meerut in 2013.He has over 20 research papers with 7 students pursuing Ph.D. under his guidance. His area ofresearch includes e-Commerce, Software Engineering, Fuzzy Logic and Data Mining. He has developed andimplemented several software projects for Defence, Govt. Organizations, and Private Organizations and in last17 years in academics he has developed various management software's for Accounts, Payroll, Time Table,Library etc.Email:dev_bbd@yahoo.com