Lessons to be learnt from students software estimation exercises

Emma Sharkey **John Paynter**

The University of Auckland
Private Bag 92019
Auckland
eshark@slingshot.co.nz,
j.paynter@auckland.ac.nz

This research has investigated the attitudes of students towards software metrics recording and accuracy. Metrics are analysed for students doing a final year Software Engineering class. Students were required to complete four group assignments, all which required the submission of metrics. A standard metrics collection template was provided for the students to complete. The metrics submitted were evaluated. The results indicated that the percentage of detailed metrics submitted was lower than that of the previous studies, despite the metrics requirements being less stringent, and the percentage of groups submitting metrics categorised as having 'no detail' being larger than that in the previous studies. A comparison between the estimated size of the software product, and the actual size (measured in lines of code (LOC)) was then conducted. The development environment used was noted, and the conversion ratio was used to convert the estimated function points into estimated LOC. In the third assignment students were required to indicate the total LOC used in the development of the prototype. The coverage of the requirements was then calculated and the actual LOC stated re-evaluated for the percentage of requirements covered in the prototype. The fact that prototypes do not incorporate to a large extent, error handling and integrity checking functionality was then taken into account. The result illustrated that the actual LOC after taking into account the requirements coverage and the omission of error handling/integrity checking functionality, was approximately three and a half times greater than the initial estimate produced. Estimation metrics for the mid semester test and final examination were collected for the purpose of evaluating the variation in function point (FP) estimations made by individuals. The method used to determine the variations in estimates in a similar study conducted by Low and Jeffery was adopted in this instance. The variation in FP estimates for the mid-semester test was 55.1%. This value was higher than that reported in Low and Jeffery (42%). The FP estimates for the group assignment were also evaluated for variation, with the variation being 153.8%. However, the variation for the final examination estimation metrics was much lower (16.5%). Low and Jeffery (1990) indicate that the 'inexperienced' function point group obtained variations of approximately 42%. For the purpose of this study, the students were considered 'inexperienced' in the application of function points as an estimation tool, but improved throughout the semester.

1. INTRODUCTION

This purpose of this paper is to explore the validity of student metrics submitted as a part of four group assignments, the test and exam for a stage three software engineering paper. This paper will evaluate the metrics submitted by students for the purpose of comparing the quality of metrics submitted. Special attention is paid to the variation within function point estimations, as this is the most commonly accepted method for sizing applications. A thorough analysis of expected lines of code versus actual lines of code will be conducted.

2. WHAT ARE SOFTWARE METRICS AND WHY ARE THEY IMPORTANT?

Software metrics have been defined as "a quantitative measure of the degree to which a system, a component, or process possesses a given attribute" (IEEE, 1993). Metrics are common amongst engineering disciplines, however, within the Software Engineering discipline, it is apparent that the collection of metrics is far less common (Pressman, 2000).

Schach (2002) indicates that software metrics are of great importance because it becomes virtually impossible to detect difficulties occurring during the development of a software product at an early stage. Difficulties addressed early are much more successfully resolved. It is indicated that if greater attention is paid to the collection and analysing of metrics, then the result is likely to be the production of a higher quality system (Schach, 2002). Thomas (1996) has indicated Software Metrics are highly

important in order to improve the quality of the Systems Development lifecycle.

Software Metrics may be categorised into product, process and quality metrics (Schach, 2002). Product metrics relate to the measurement of the software product, Process metrics relate to the actual software process itself and Quality metrics relate to the identification of areas within the software development process that are not performing to a level that they should be (Burgess, 1997).

3. METHODOLOGY

As a part of the course assessment for INFOSYS 332 (Software Engineering) at the University of Auckland, students are required to complete four group assignments, based on a case study (property investment).

3.1 The process in which we collected metrics from students

A metrics collection template was supplied with every assignment, for describing and then collecting the pre-defined metrics and their units. The templates used differed from that used in Thomas (1996) and Burgess (1997). Metrics were collected from eight groups. Between 5 and 10 percent of the mark for each assignment were allocated towards software metrics (Table 1), thus giving motivation towards the submission of accurate metrics.

3.2 Evaluation of Student Metrics Quality

The categories illustrated in Table 2 will be utilised for the purpose of assigning a level of 'quality' to the student metrics submitted, adapted from the previous studies. The software metrics submitted will be categorised into the perceived level of effort placed in ensuring the accuracy, thus mimicking the approach taken in Thomas (1996) and Burgess (1997).

Statistical analysis will be applied to determine differences between the four assignments. The author believes this mark deduction approach accurately reflects metric quality.

As an illustration, the Product metrics submitted in the example assignment gathered in the requirements phase shown in Table 3 will be taken as being correct. However, if it is apparent that an error has been made, then adjustments can be made through referencing other assignment components.

Table 1 - Group Assignment Metrics Mark Allocation

Assignment	Description	Marks Awarded (out of 100)
One	SRS, SPMP	5
Two	Functions (UML)	7
Three	Prototype	5
Four	Testing and Documentation	10

Table 2 – Metrics Classification Categories

Classification	Mark Deduction
Detailed	0.0
Attempted	0.5
Combined	1.0
Half-Hearted	1.5
Concocted	2.0
No Detail	2.5 or greater

Table 3: Product Metrics Submission from Group X, Assignment 1.

Description	Measurement	Units
Number of functional requirements	9	Count
Number of non-functional requirements	8	Count
Size of SRS	10/1818	Pages/words
Size of SPMP	23/4684	Pages/words
Activities in SPMP	5	Count
Entities	20	Count
Function Points (UFP)	55.44	Count
Technical Coefficient	0.9	
Adjusted Function Points	49.896	Count

Table 4 – Metrics Classifications for individual group assignments

Classification	Assn 1	Assn 2	Assn 3	Assn 4	Overall	Overall %
Detailed	1	1	0	0	2	6.3%
Attempted	0	3	3	2	8	25.0%
Combined	3	3	3	3	12	37.5%
Half-Hearted	2	0	0	1	3	9.4%
Concocted	1	0	2	0	3	9.4%
No Detail	1	1	0	2	4	12.5%

3.3 An Analysis of Software Estimation

For the estimation of FP, students followed one of two methods in the estimating of the total number of Unadjusted Function Points (UFP) – the method proposed by Albrecht (1979) or that of Symons (1986). The technical complexity factor is then calculated.

3.3.1 An Analysis of Software Estimation – Group Assignment Metrics

Students were required to indicate the total number of function points for the Property Investors System. Variations amongst the estimated total number of function points between the eight groups will be compared to the variations amongst size estimations made by other studies. The level of quality and relevance of the metrics submitted was judged at the time of marking.

Both the UFP and the technical complexity factor were recorded in the metrics template. The total number of adjusted function points (AFP) was then calculated and recorded in the metrics template. In the third assignment, the students were required to stipulate the total number of lines of code that had been produced. This, along with the development environment utilised was recorded. The AFP specified in the metrics template for the first assignment will then be converted to an estimated number of lines of code in order to gain insight into the level of estimation accuracy achieved by the students in this assignment. The fact that the students only submitted prototypes will also be taken into account, through adjustments made for the coverage in requirements implemented in the prototype and the omission of error handling and integrity checking code.

3.3.2 An Analysis of Software Estimation – Test and Examination Estimation Metrics

As a part of the mid-semester test, the students were required to produce an estimate for the case specified. The case for the test was based upon the development of the Virtual NPC website (www.virtualnpc.co.nz). The Virtual NPC website is an online competition based upon Rugby's National Provincial Championships (NPC).

For the final examination, the students were again required to produce an estimate for development of a web site for booking holiday accommodation. This case was in a similar domain to that of the group assignment (Property Investment - residential tenancies).

Low and Jeffery (1990) conducted a study comparing the variation in Function Point estimations between 'experienced' and 'inexperienced' individuals in the calculation of Function Points. This study calculated the variance by dividing the standard deviation into the mean function point calculation gained – thus gaining a unit-less measure that may be applied to other studies. Their results illustrated that the 'experienced' group had a variance in estimations of 21.6%, with the two inexperienced groups having variances of 41.2% and 42% respectively. . Such findings will be applied in this study.

4. RESULTS OF THE METRICS ANALYSIS

4.1 Quality of Student Metrics

As outlined in Table 2, the quality of the metrics were analysed following the guidelines stipulated in Thomas (1996) and Burgess (1997) (Table 4).

4.2 Analysis of Group Assignment Estimation Metrics

Table 5 illustrates the total number of UFP, the TCF and the total number of AFP obtained from

Table 5 - Student Estimation Metrics

Group	UFP	TCA	AFP	Development Tool	Actual LOC
A	68	1.05	71.4	MS Visual Studio.net	2450
B	82	1.05	80.02	MS Access	1120
C	1018	1.14	1160.5	MS Visual Studio.net	345
D	248	0.97	240.6	MS Access	2484
E	55.4	0.90	49.9	MS Visual Studio.net	750
F	60	0.95	57.0	MS Visual Studio.net	875
G	181	1.00	181.0	MS Access	403
H	126	0.92	116.0	MS Visual Studio.net	1232
Mean:	**229.8**	**0.998**	**244.5**		**1207**

Table 6 - Estimated FP/Estimated LOC

Group	Estimated FP	Estimated LOC	Actual LOC
A	72	1428	2450
B	80	1201	1120
C	1161	23211	345
D	241	3609	2484
E	50	998	750
F	57	1140	875
G	181	2715	403
H	116	2320	1232
Mean:	**245.0**	**4578.0**	**1207**
SD		**7584.23**	**836.02**

the estimation conducted for the first group assignment. The author calculated the total number of adjusted function points by utilising the following formula:

AFP = UFP * TCF

The table illustrates that the mean UFP estimation was 229.8, with the mean TCF being 0.998, and the mean AFP being 244.5.

The development tools used in the prototype were recorded, shown in Table 5. The purpose of this is to, as accurately as possible, estimate the total LOC that the groups anticipated the software product would require.

Group C had a UFP value (1018), much greater than the estimates made by other groups. This would have had the effect of increasing the mean value to an extent not reflected in a number of the other groups' estimates.

Table 5 illustrates the total LOC stated by each group after the development of the prototype. The mean actual LOC as stated in the student assignments was 1207. There was significant variance in the actual LOC stipulated by the groups. The standard deviation of the LOC above was 836, representing approximately 70% of the mean actual LOC.

Lines of Code per Function Point

The eight groups used a total of two development tools. For the purposes of obtaining the appropriate LOC per FP measurement, MS Access is going to be considered a 'code generating tool' and MS Visual Studio.NET will be classified as a 4GL.

The values obtained for the LOC per FP were obtained from Griffith University (http://www.int.gu.edu.au/courses/2203int/mm.html) of 15 and 20 LOC per FP for Access and Visual Studio.net respectively.

Using the LOC per FP conversions as illustrated, the expected number of LOC was calculated based upon the estimated number of FP students supplied in the first assignment.

Table 8 – Requirements Coverage in the Prototype

Group	Functional Requirements		Coverage (%)	Weighted LOC
	Planned	Implemented		
A	12	9	75.0	3267
B	14	6	42.6	2629
C	12	5	41.7	827
D	14	6	42.9	5790
E	9	4	44.4	1689
F	13	5	38.5	2273
G	11	11	100.0	403
H	14	6	42.9	2872
Mean:	**12.4**	**6.5**	**53.5**	**2469**

Table 7 – Estimated/Actual LOC Confidence Intervals

	Sig. (2-tailed)	95% Confidence Interval of the Difference	
		Lower	Upper
Estimated LOC	0.132	1007	5629
Actual LOC	0.005	508	1906

Table 6 illustrates the estimated FP and LOC obtained from the estimation made in assignment one. The estimated LOC for the assignment case is 4578 LOC.

A comparison will now be conducted between the estimated LOC and the actual LOC stated by students

Table 7 illustrates the estimated LOC is approximately four times greater than the actual LOC.

The 95% confidence interval obtained for the actual LOC was between 508 and 1906 lines of code. The 95% confidence interval obtained for the estimated LOC is between 1007 and 5623.

Students produced a prototype, and thus a large proportion of the code has been omitted. This is because a prototype will not incorporate extensive error handling code, as would a fully developed system. In assignment 4 the students were required to produce a traceability matrix detailing the number of functional requirements the prototype implemented. This will now be examined and the actual LOC will be adjusted for each group based upon the proportion of functional requirements they implemented. The estimated LOC is based upon all functional requirements specified, thus, it appears that an adjustment to the actual LOC specified is required.

Table 8 indicates of the number and proportion of functional requirements that each group implemented in their prototype. The coverage percentage obtained will be used in order to convert the actual LOC to a value that will more closely resemble the full set of functional requirements that would have to be implemented in the final system.

On average, after a weighting had been applied, the actual LOC approximately doubled in value. This is illustrated in Table 8. The new 'weighted LOC' figured produced takes into account the requirements specified but not implemented.

Table 9 illustrates that the difference between the estimated LOC and the weighted LOC has decreased due to taking into account the coverage (the proportion of functional requirements covered in the prototype submitted). The standard deviation, however, has increased from 836 to 1667.6, indicating an increase in the variability of the actual LOC.

Table 9 – Comparison between estimated and coverage adjusted LOC

Group	Estimated LOC	Weighted LOC
A	1428	3267
B	1201	2629
C	23211	827
D	3609	5790
E	998	1689
F	1140	2273
G	2715	403
H	2320	2872
Mean:	**4578.0**	**2469**
SD	7584.23	1667.56

Table 13 – Expected LOC Statistics

	Minimum	Maximum	Mean	SD
Expected LOC	2015	28950	12344	8337.8

Table 10 – Estimated/Weighted LOC Confidence Intervals

	95% Confidence Interval of the Difference	
	Lower	**Upper**
Estimated LOC	1007	5623
Weighted LOC	1075	3863

Table 11 – Comparison between Estimated and Expected LOC

Group	Estimated LOC	Expected LOC
A	1428	16335
B	1201	13145
C	23211	4135
D	3609	28950
E	998	8445
F	1140	11365
G	2715	2015
H	2320	14360
Mean:	**4578.0**	**12344**
SD:	7584.23	8337.78

Table 10 illustrates that the 95% confidence interval for the weighted LOC value is between 1075 and 3863 LOC.

Prototypes were developed, thus the weighted LOC values will not take into account the fact that error handling and validity checking had not been implemented. From experience, approximately 20% of the LOC is allocated towards functionality, with the remaining 80% being allocated to error handling for a fourth generation language, indicating further FP manipulation is required. The results are illustrated in Table 11.

Table 12 – Expected/Estimated LOC Confidence Intervals

	95% Confidence Interval of the Difference	
	Lower	**Upper**
Expected LOC	5373	19314
Estimated LOC	1007	5623

After accounting for the lack of error handling code in the prototype developed by the students, the mean LOC for the prototype is 12,344. This is approximately three times greater than the mean estimated LOC by the students.

Table 12 shows the 95% confidence interval for the mean expected LOC was between 5373 and 19314, with the estimated LOC being between 0 and 10,918 LOC. The two confidence intervals overlap, mainly due to the large degree of variance associated with the expected LOC. Table 13 illustrates a standard deviation of 8338, which is approximately two-thirds the size of the mean expected LOC value obtained.

4.3 Analysis of Test/Exam Estimation Metrics

The number of UFP, TCA, the stipulated LOC per Function Point, LOC per hour, and total effort (measured in person-hours) was collected from the mid-semester test and final exam scripts. The relevant statistics are illustrated in Table 14.

Table 14 shows the terms test statistics and show a large degree in variability within estimates. The instructor's estimation is also shown. The minimum number of UFP stated was 42, the maximum UFP stated was 645, with the mean UFP value stated being 207. This shows a large variation in student FP estimates.

The examination statistics are shown in Table 15, and illustrates a smaller degree of variability between individual estimates than the terms test. The instructor's estimation is also stated. The difference between the minimum and maximum UFP for the ex-

Table 14 - Terms test estimation metrics

	Minimum	Maximum	Mean	SD	Variance	Instructor
UFP	42.0	645.0	209.7	106	11226.88	282
TCA	0.92	1.15	1.04	0.06	0	1.02
LOC per FP	0.90	60	13.1	10.41	108.4	20
LOC per HR	5	277	31.5	65.97	4352.63	10
Effort (p/h)	30	2341	294.6	428.16	183323.51	576
AFP	43.7	696.6	220.2	118.65	14076.97	288

Table 15 - Examination Estimation Metrics

	Minimum	Maximum	Mean	SD	Variance	Instructor
UFP	170	315	234.6	36.59	1338.89	304
TCA	0.92	1.15	1.02	0.06	0	1.00
LOC per FP	10	80	14.6	13.23	174.97	10
LOC per HR	5	20	10.1	2.42	5.88	10
Effort (p/h)	174	1124	335.1	194.13	37688.1	304
AFP	173.8	318.2	239.9	39.69	1575.66	304

Table 16 – FP Estimation Variation

FP Source	AFP	SD	Percentage Ratio
Group Assignment	244.5	376.01	153.8%
Mid Semester Test	217.8	118.65	54.5%
Final Examination	239.9	39.69	16.5%

amination was 145, with the terms test having a 603 difference. The mean estimated UFP in this instance was 234.6, with the mean effort (person-hours) being 335.1 person-hours.

The variations in function point (AFP) estimations obtained in this study will now be compared with that of Low and Jeffery (1990). Students had not used FPs before this course, and are considered 'inexperienced'. Table 16 illustrates the mean and standard deviation of the FP estimations made by students in this study, showing a decrease in variability over the semester.

In both instances, the mean UFP and AFP, as calculated by the students has been lower than that of the instructor, as students tend to underestimate the size of the software product to be developed.

5. DISCUSSION

5.1 Evaluation of Student Metrics Quality Discussion

For the purpose of the comparison between this study and the studies conducted by Burgess (1997) and Thomas (1996), each classification was summed across the four assignments, and compared with the percentages obtained in previous studies. This is illustrated in Table 17. What becomes apparent is that the percentage of students submitting 'detailed' metrics is much lower than the percentage of students submitting 'detailed' metrics in the previous studies, potentially due to different approaches used in the different studies. This approach only requires students to fill in a collection form, as opposed to the detailed logs completed in the previous studies.

One would expect the percentage of detailed logs submitted in this study would be higher than that of previous studies due to the lower complexity of the metrics templates used in this study. In addition this study involved final (undergraduate) year students whereas the previous studies were of second year students.

5.2 Group Assignment Estimation Metrics Discussion

The difference in the estimated and actual LOC values is indicative of the shortcomings of such an estimation method. Schach indicates that studies

Table 17 – Comparison with Results of Previous Studies

Classification	This Study	Thomas (1996)	Burgess (1997)
Detailed	6.3%	21.0%	31.5%
Attempted	25.0%	18.4%	30.3%
Combined	37.5%	34.2%	23.0%
Half-Hearted	9.4%	18.4%	8.4%
Concocted	9.4%	2.6%	3.4%
No Detail	12.5%	5.2%	3.4%

have shown that the difference between estimated and actual software size (using FP as an estimation method) to be in the realm of 200%. This study has indicated a difference of approximately 350% between the estimated and actual outcome. Perhaps the relative inexperience by the students in using function points as an estimation method impacted upon the estimation values, or inaccurate indication of the LOC expended in the development of the prototype, with some potential bias being introduced due to FP conversions made.

5.3 Test/Examination Estimation Metrics Discussion

The results obtained illustrate that the percentage ratios are higher than that found in the study of Low and Jeffery (1990). The percentage ratio for the group assignment was 153.8%, 54.5% for the mid-semester test and only 16.5% for the final examination. The sample size for the mid semester test and exam is approximately three and a half times larger than that of the group assignment (i.e. they are done individually). The discrepancies (with respect to the group assignment and mid semester test estimates) could be due to the inexperience, relative to the 'inexperienced' group in the study of Low and Jeffery (1990), or lack of access to the resources provided to participants in the previous study. However, it may potentially give an indication as to the vast variations in function point estimates made by different individuals.

The final examination illustrated a dramatic reduction in the variation in function point measures. This could potentially be due to students gaining more experience in the conducting of function point estimations.

6. CONCLUSION

This study has made comparisons with the results of previous studies conducted. This study has illustrated the great degree of variability in the function point estimates made, much more in some instances than the previous studies illustrated.

The analysis of the lines of code used in the development of the prototype initially illustrated that the actual LOC was lower than that of the estimation. However, once the fact that the students only submitted a prototype was taken into account, the actual LOC was approximately four times the value of the estimated LOC.

The quality of the student metrics was contrasted against those of different studies. Similar rating criteria were used to that of the two previous studies. The overall quality of student metrics appeared lower in this study than the level in the previous two studies.

6.1 Limitations of this research

It would have been beneficial if the data set of student metrics collected was larger. If greater time had been available, then this template approach towards the collection of metrics would have been conducted over a time period exceeding one semester.

REFERENCES

Burgess, M. *"Software Metrics: An Experiment in Student Metrics Collection"* 1997, Unpublished Honours Dissertation.

Low, G and Jeffery, D. *"Function Points in the Estimation and Evaluation of the Software Process"* IEEE Transactions on Software Engineering, Vol. 16, No. 1, January 1990.

Pressman, R. (2001) ."*Software Engineering, A Practitioners Approach"* 2000. McGraw Hill.

Schach, S. *"Object-Oriented and Classical Software Engineering"* 2002, McGraw Hill, ISBN 0-07-112263-X.

Thomas, R. *"A Practical Experiment in Teaching Software Engineering Metrics"* 1996, Proceedings of the International Conference on Software Engineering: Education and Practice, IEEE Computer Press, Los Alamitos, CA, pp 226 – 232.