EMPIRICAL INVESTIGATION OF THE EFFECT OF THE LINES OF CODE ON FUNCTION POINT IN EMBEDDED SOFTWARE

Shalley Sharma[1] , Arvind Kalia[2]

[1&2] Professor, Department of Computer Science, H.P. University, Shimla (H.P.)
babylucky.shalley40@gmail.com, arvkalia@gmail.com

ABSTRACT: *The embedded software has grown complex and pervasive enough to attract the attention of the computer scientists and the researchers. As the embedded software market has grown in the last decade and will continue to grow in the coming years. The embedded software in majority are meant for performing in real time constraints in robust manner. So this makes the researchers to study the source code of embedded software and find out the relationship between the Lines of code and the function points in the embedded software. As the function points is supposed to be the predictors of the complexity of the software so the relationship between the Function Point and Lines of code can provide an insight to the developer to produce more reliable and less complex embedded software. For the empirical study the five versions of the embedded software namely gSOAP were taken. Then the collected data is analyzed using a source code analyzer Resource Standard Metrics (RSM) to get the data values for Lines of code and Function Point. Result shows that the Function Points and Lines of code has a positive relationship which means that as the Lines of code increases Function point also increases. However further empirical study are needed before the results can be generalized.*

Keywords: *Embedded Software, Resource Standard Metrics (RSM) Tool, Function Point.*

© 2011 Journal Anu Books

I. Introduction

Embedded software is computer software that plays an integral role in the information technology. Embedded software's principal role is not information technology, but rather the interaction with the physical world. Over the last 20 years, software's impact on embedded system functionality, as well as on the innovation and differentiation potential of new products has grown rapidly. This has led to an enormous increase in software complexity, shorter innovation cycle times, and an ever-growing demand for extra-functional requirements—software safety, reliability, and timeliness [1].

Embedded software is usually written for special purpose hardware that is computer chips that are different from general purpose CPUs, sometimes using real time operating system such as fusion RTOS, nucleus RTOS, Integrity, OSE, Linux, eCos.[7] This type of software is placed in read-only-memory(ROM) of the product and control the various functions of the product. The product could be an aircraft, automobile, security system, and signaling system. The embedded software handles hardware components and is also termed as intelligent software [2]. The embedded-software market has grown swiftly in the last decade and will continue to do so in coming years. Embedded software's specific properties, such as hardware dependencies, make its development different from non-embedded software. So, we might expect that this domain would employ specific software development technologies.

The complexity of the embedded applications and consequent size of their programs is growing rapidly. Their devices now often sit on a network, wireless or wired. And the applications are getting much more dynamic, with downloadable customization and migrating code. Meanwhile, reliability standards for embedded software remain very high, unlike general-purpose software [8]. So it becomes the need of the hour that an analysis of the source code for embedded system is done to have an idea that how the development of such kind of the software can be improved in the terms of complexity and reliability. The function points in the source code are supposed to be the indicator of the complexity. So having an idea of the relationship between the lines of code and function points can provide a

© 2011 Journal Anu Books

better insight to the developers of the embedded software to develop less complex and more reliable software.

A. Lines of code (LOC)

Conte [3] has defined lines of code as "lines of code is any line of program text that is not a comment or blank line, regardless of the number of statements or fragments of statements on the line. This specifically includes all lines containing program header, declarations, and executable and non-executable statements".

Since lines of code (LOC) only measures the volume of code, one can only use it to compare or estimate projects that use the same language, and is coded using the same coding standards. To change one is to change the volume of code. A better method to compare without regard to direct volume is to measure the complexity of the software. This can be done with function point analysis, which can measure the complexity of the programs inputs and outputs. The lower LOC measurement is the better [9].

B. Function Point (FP)

Function point approach is independent of the language, tools, or methodologies used for implementation; i.e., they do not take in to consideration programming language, processing hardware. Function points can be estimated from requirement specification or design specifications, thus making it possible to estimate, development effort in early phase of development. Function points are directly linked to the statement of requirements, any change of requirements can easily be followed by a re-estimate [4]. Function points are based on the system user's external view of the system, non-technical users of the software system have a better understanding of what function points are measuring. This method resolves many of the inconsistencies that arisen when using lines of code as a software size measure.

II. Related Work

"Marcio F. S. Oliveira "In their research paper" Software quality metrics and their impact on embedded software" discussed that the embedded system analyze the correlation between these metrics and highlight the

© 2011 Journal Anu Books

behavior for quality and physical metrics, which help us to better understand the trade-off between reuse and optimization.

In this work, we found that the embedded software design cycle can greatly take advantage from the use of software product quality metrics. These metrics can help to raise the reuse factor and as consequence shrink time-to market.[10]"Ulisses Brisolara Corrêa, "In their research paper" Towards estimating physical properties of embedded systems using software quality metrics " discussed the complexity of embedded devices poses new challenges to embedded software development in addition to the traditional physical requirements. Therefore, the evaluation of the quality of embedded software and its impact on these traditional properties becomes increasingly relevant.[5]"Jiang In their research paper "Design of A General Embedded Software Debug System "discussed with the extensive application of embedded systems, embedded software development becomes very important, and debugging is an indispensable stage in the software development process.

In early stage of embedded system development, the embedded debugging system provides program download, debugging and other functions, and production it provides the image file download function.

The purpose of this paper is to carry out requirements analysis for the embedded debugging system functions, build a powerful, user-friendly, easy-to-transplant embedded debugging system and give solutions for the several key technologies to realize it.[6]"Hans Henrik LGvengreen In their research paper**,** "Design of embedded, real-time systems: Developing a method for practical software engineering" discussed the methodological issues and practical problems in development and industrial use of a theory-based design method for embedded, real-time systems are discussed.

Our main contribution has been to delineate an order in a technique for deriving states and operations, and systematic checks of a design with respect to system requirements.

File system reliability has always been a strong concern in the embedded operating system. This reliability frequently involves a choice between raw performance and integrity guarantees.[11]

© 2011 Journal Anu Books

III. Research Methodology

In this study the five versions of the gSOAP embedded software [6] were considered for the investigation. Then the source code of these five versions of gSOAP was analyzed using the Resource Standard Metrics (RSM)[13]. The Resource Standard Metrics (RSM) was used to collect the data about two software parameters namely Lines of Code (LOC) and Function Point(FP). Resource Standard Metrics tool works with most operating systems and it allows users to examine, find the source code for and run quality tests on a number of source codes, including C, C++, C# and Java. Users can run tests on metric and then compare those tests to the baseline code, helping teams determine whether or not they are in control of the process. Other features include the ability to run statistical analysis on source codes, and the program can even change Disk Operating System (DOS) code into the UNIX format (a type of operating system that has been around since the 1960s). Resource Standard Metrics can generate Extensible Markup Language (XML) and Extensible Style sheet Language(XSL), Microsoft Excel file) reports [14].

IV. Results

Table 1 shows the data values of LOC and FP of gSOAP for five different versions which are generated by RSM.

Table 1: Data Values for LOC & FP of Different Versions of gSOAP

	VERSION (V1)	VERSION (V2)	VERSION (V3)	VERSION (V4)	VERSION (V5)
LOC	25732	32145	32508	53002	104474
Function Point	485.5	606.5	613.4	1000	1971.7

To represent the data diagrammatically in the appropriate manner, the values are presented after dividing by a constant value. Figures 1 below indicate a positive relationship between the LOC and the FP. Even the correlation coefficient computed for the above data values shows the positive correlation between the LOC and FP. The correlation coefficient computed

© 2011 Journal Anu Books

for the above data values was 0.9999999894424294 which is a clear indication of the positive correlation.

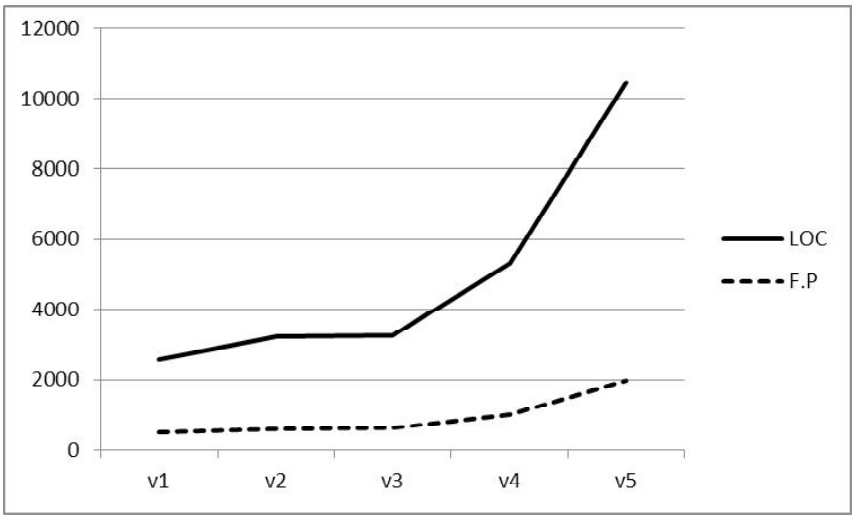

Fig. 1 LOC vs FP for Gsoap

V. Conclusion and Future Work

The researchers started the research with the aim of studying the effect of the LOC on the function point complexity. As software function point is supposed to be the indicator of software complexity, and reliability so the correlation between the LOC and the function point can provide a feedback to the developers of the embedded software for making the software more reliable and less complex. We collected sufficient and latest data at various levels to verify this relationship. It is evident from the analysis that the LOC and FP has a positive relationship which means that as the LOC increases then the function points also increases and hence the complexity. However further empirical studies are needed before these results can be generalized. The purpose of this study is to analyze the correlation of the selected quality metrics in the embedded software.

© 2011 Journal Anu Books

References

[1] Peter Liggesmeyer, Mario Trapp, "Software Engineering Trends in Embedded Software Engineering", Software, IEEE , Volume 26, Issue 3 ,May-June 2009 pages 19 - 25 , ISSN: 0740-7459

[2] K.K. AggarwaL, Yogesh Singh, "Software Engineering", New Age International Publishers, page130-135, 2005.

[3] Conte S.K.,Dunsmore H.E., Shen V.Y., "Software Engineering Metrics and Models", The Benjamin/Cummings Publishing Corporation Inc., California, USA, 1986.

[4] Ince D., "Software Metrics: Measurement for Software Control and Assurance", New York: Elsevier, 1989.

[5] Ulisses Brisolara Correa, Luis Lamb, Luigi Cargo, Lisane Brisolara Julio Mattos, Towards estimating physical properities of mbedded system using software quality metrices, cit, pp.2381-2386,2010 10th IEEE International conference on computer and information technology,2010

[6] Jiang, Chengyan,Wu, Siyuan,"design of a general embedded software debug system" Chongqing Electric Power College, Chongqing 400053,China;2009

[7] http://en.wikipedia.org/wiki/Embedded_software accessed on 31-07-2011 at 11:30 a.m

[8] http:\\www.ptolemy.eecs.berkeley.edu/publications/papers/02/embsoft/embsoftware.pdf accessed on 14-08-2011 at 5:00pm

[9] http://c2.com/cgi/wiki?lines of code accessed on 04-08-2011 at 2:25 p.m

[10] http://unipaderborn.academia.edu/MarcioOliveira/Papers/654602/Software_quality_metrics_and_their_impact_on_embedded_software accessed on 26-06-2011 at 2:00pm

[11] http://orbit,dtu.dk/getResource?recordid=200208&objectid=1&versionid.pdf accessed on 02-07-2011 at 9:00am

© 2011 Journal Anu Books

[12] http://sourgeforge.net/projects/gsoap2/files/gsoawin322.8.zip/download=download2.8versio n accessed on 23-08-2011 at 3:30 p.m

[13] http://sourceforge.net/rsm.zip/msquaredtechnologies.com accessed on 23-06-2011 at 4:57 p.m

[14] http://www.ehow.com/list_6948597_software-quality-metrics-tools.htmlacceseed on 23-07- 2011 at 4:10 p.m

© 2011 Journal Anu Books