A Simplified Alternate Approach to Estimate Software Size of Startups

Chandrasekaran Sridharan
Department of Computer Science and Engineering,
Thiagarajar College of Engineering, India
cscse@tce.edu
(Corresponding Author)

Sudhaman Parthasarathy
Department of Applied Mathematics and Computational Science,
Thiagarajar College of Engineering, India
parthatce@gmail.com

Abstract: *This paper proposes an alternate approach to startups to estimate the size of software product to be built by them using the Software Product Points (SPP). Dataset from 20 software projects of a startup company in India was used to validate the proposed approach and learn lessons out of it. The estimated software product points and the project efforts were found to have a strong positive correlation, thereby indicating the suitability of the proposed approach for its utility by the managers of future software projects of startups. We also briefly outline the implications for project managers of startups and scope for future research.*

Keywords: *Software size, software projects, quality function deployment, process improvement, optimization*.

Received January 19, 2021; accepted September 5, 2021
https://doi.org/10.34028/iajit/19/4/12

1. Introduction

Software startups are small companies that are well known for their competency to develop and deliver innovative products at the earliest possible time [6, 12]. The high failure rates in software startups could be due to uncertainty in market conditions, skewed business cycles, poor estimation of software size or inaccurate estimation of project efforts required for their projects [12]. It is very much essential to estimate the project efforts accurately at the initial stage of the project itself either by using simulation or by means of utilizing the project repository of the respective software companies [11]. Adequate software engineering based research for startups is essential as previous research endorses that the success of projects of these startups are characterized by their resources (project efforts and other IT infrastructure), product engineering, ability to meet software as well as market requirements [9, 10, 11, 12, 13, 14]. In the software engineering terminology, the term ‘project efforts,’ refers to the manpower required for the development of the software product in a software project which is measured normally in person-hours, person-day or person-months [4]. The Function Point (FP) is a unit of measurement to represent the core functionality expected by the end user from the software product. The FP usually represent software size in a software project. The Lines of Code (LOC) is captured by counting the number of lines in the text of the program's source code. Previous research [22] suggests that the FP’s or the LOC’s based software size measurement process grows proportionate to the size of the software project. Hence, in this paper, in our proposed approach, we measure the software size of startups using the Software Product Points (SPP) instead of the FP and the LOC.

We define the term SPP as a specialized unit of measure for the size of a software product. The Software Product Points are calculated by integrating the Market Points (MP) and the Product Points (PP). The market points are estimated based on market requirements for the software product to be built by startups and its underlying complexity/constraints. The product points are estimated based on the user stories gathered from the customer and the subsequently defined product requirements. Software size measurement and project efforts estimation still remains a topic of discussion in startups from the perspective of software engineering research [13, 17, 18]. Our objective is to suggest an alternate approach to the managers of startups engaged in software projects that would certainly help them

1. Achieve a deeper understanding of the software size measurement in startups ecosystem.
2. Estimate project effort as accurately as possible.
3. Gain insights on how software size and project efforts estimation in startups account for efficiency of software projects.

The remainder of this paper is organized as follows. In section 2, we provide a brief background on the need for estimation of project efforts based on software product points by start-ups and also related research works to outline the limitations in the traditional methods for such measurements by startups. We present our proposed approach in section 3. Implementation of our proposed approach is explained in section 4. Section 5 presents a discussion on obtained results. Section 6 briefs the contributions to research and their implications for practice. Section 7 brings out our concluding remarks.

2. Background and Related Work

This section provides background of traditional methods on software size and effort's estimation in software projects and summarizes related research works emphasizing the need for accurate software size estimation of software projects of startups.

2.1. Need for an Alternate Approach to Measure Software Size of Projects of Startups

The primary reasons could be the fact that these managers could not embed startup's specific factors such as the market requirements, constraints or challenges associated with these requirements and the user specific product requirements/features into the traditional methods used by them presently during software size measurement process [18]. Software engineering researchers acknowledge the need for better process models, and software size measurement methods of projects of software startups which in turn would lead to a better project effort's estimation at an early stage of the project itself [12]. Understanding startups' software engineering context will enable development of software engineering practices that are more specific to startups [18].

2.2. Traditional Methods for Software Size Measurement

Previous research studies [7, 9, 12, 15, 16] on software development of startup companies have time and again indicated that the software engineering methods and practices for traditional IT services companies could not be adopted right away by software startup companies for various reasons such as resource constraints, market conditions, low cost software product development and untrained man-power.

Effort's estimation during software projects is solely depending on the accuracy of estimate size of the software product to be built. Extensive research has gone into this specific area of efforts estimation by software engineering researchers in the past two decades. All these methods or models are found to use the lines of code or the function points which may not suit the software projects of software startups. We aim to fill this gap in the literature and propose a unique and an alternate approach for software size measurement of projects of startups in the next section.

3. Proposed Approach

In this section, we present our proposed approach to estimate the Software Product Points based Software Size estimation (SPPSS) for software startups. Software startup companies are meant to develop innovative software products that reach out to a larger market than a traditional product that would be preferred by a specific client [3]. Such practices of startups require a different approach to elucidate the requirements and estimate the size of the software and the proportionate efforts required for the software project.

We observe that the existing methods or models used by traditional software services companies assume bottom-up thinking where the software product is viewed only from the point of its core functionalities. Such functionalities would yield the various functional and non-functional requirements of the software product and are basically used to quantify the size of the software product. This leaves a gap in the startup ecosystem and we seek to fill this gap through the present research work.

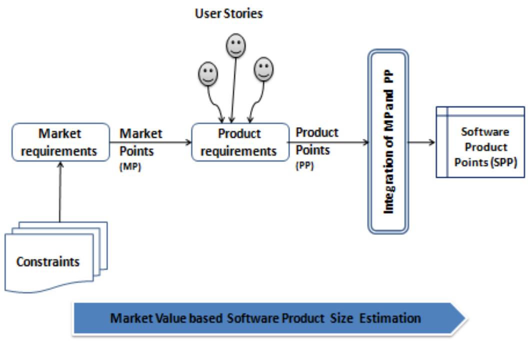

Figure 1. Software product points estimation: a high-level view.

In this paper, we propose a unique and an alternate approach, referred as estimation SPPSS, to estimate the size of the software product to be built by a startup company. The steps involved in the "SPPSS" are shown in the Figure 1. From the Figure 1, we find that, as a first step, the Market Requirements (MR) are captured by the product development team of a startup. These requirements take into account the constraints that could encompass the market requirements.

The market requirements are meant to primarily reflect the user view of the problem to be solved by the software product of a startup. The product requirements (PR) are then extracted from the market requirements considering the user stories captured from the customer's site. The product requirements are meant to outline the functional solution to the market requirements. The product requirements are expected to turn the vision of a product into reality by outlining what precisely the engineers are supposed to build. These PRs would comprise of the functional and non-functional requirements required for the product from user's perspective. The proposed approach "SPPSS" requires the estimation of SPP using MP and the PP. We define the term MP as the summation of the number of constraints that circumvents a given market requirement. Such constraints will vary from one market requirement to another. These constraints are meant to indicate the limitations of the proposed software solution by a startup. They may also be induced by the demand for features such as multi-tasking, remote location access, low power consumption and mobility. Hence, these constraints that are associated with every market requirement are rated during the calculation of market points and we denote this as "C_i" for a given market requirement, say 'MR_i'. The PP is defined as the summation of the number of tasks/processes required to implement a given product requirement. Now, the integration of the market points and the product points yield the SPP. The computed value of the SPP denotes the size of the software product to be developed by a startup. We now present mathematically, the definition of the SPP as follows.

Software Product Points (SPP) = $\sum$ [1+(C_i/100)]*PP, (1)

i=1

Where N=Number of Market Requirements (MR)

C_i =Number of constraints associated with a given MR_i

N

Here, $\sum$ [1 + (C_i / 100)] is referred to as "Market Points (MP)"

i=1

N M

Also, Product Points (i.e.) PP = $\sum \sum P_{i\,j}$

i=1 j=1

Where M = Number of product requirements (PRs) for a given MR and

P_{ij}=Number of tasks/processes required to implement a given product requirement.

4. Implementation

In this section, at first, we elaborate on the research method we have chosen to implement our proposed approach "SPPSS". Then, we describe the dataset used in this research work for the application of the "SPPSS". Finally, we present the results and lessons learned from it.

The Case Study research method proposed by Yin [21] has been used in this paper. We executed the various steps outlined in the Figure 1 as suggested by Yin [21]. Such case study method based validation of newly designed methods has been well recognized by the research methodologists in the field of software engineering [8, 13]. Basically, Yin [21] suggests that case study research is well suited for scenarios when a 'how' or 'why' question is being examined upon some existing models or methods, previous research works by Daneva [8] and Wohlin *et al*. [20] argue that the case study research also facilitates us to seek a solution to the research problem that is limited to 'which is better' question [19, 20]. Hence, we decided to choose the case study research based validation of our proposed approach "SPPSS".

To implement our proposed approach, we interfaced with a software startup company (pseudoname "Tenic") headquartered in India and operate with their customers in United States and United Kingdom for their software product development. We were supported by two technical members involved in the software projects of the startup "Tenic" to provide necessary data for execution of the proposed approach "SPPSS (please refer to Figuer 1). With the support of these members, we initially recorded all the market requirements and the product requirements using a Microsoft Excel sheet. Data that were recorded in the Excel sheets was taken as input and the computation of the MP and the Product Points were carried out to estimate the SPP which in turn reflects the software product size of startups. The Case Study Company 'Tenic' is a software startup company which has over 9,000 users from 85-plus customers across the globe. Their employee strength would range somewhere between 35 and 45 on an average in the past three years. The company is 8 years old and is well recognized by their customers across the globe for their customer centric software solutions. They had a dedicated team of members for software development involved in managing their projects. However, this company had been facing problems in developing the market driven software products in the recent past. Hence there was also a decline in their annual revenue during 2016-2018. Their average annual revenue is nearly $27,000 in the past 5 years.

4.1. Dataset

We have used dataset of 20 software projects that were undertaken by our case study company "Tenic" during April 2014 to May 2018. We have applied our approach "SPPSS" to these 20 projects of Tenic, each of which deals with the development of a software product for a client organization. These projects had the following issues:

1. Schedule slippage.
2. Deviation between the estimated project efforts and actual efforts used for these projects.
3. Huge overrun in the estimated project costs.

We decided to choose these 20 software projects with such characteristics for our case study, as we want to explore the impact of variation in the software size (i.e.,) inaccurate software size measurement at the initial stage of the project and the actual software size at the end of the project. Such deviations would be resulting in inaccurate project effort's estimation in these projects. Table 1 shows the descriptive statistics of 20 software projects.

Table 1. Descriptive statistics of 20 software projects (P_i) of startup 'Tenic'.

Variables	**N**	**SUM**	**Standard Deviation**	**Minimum**	**Maximum**	**Mean**
Actual Efforts	20	170.83	5.63	1.16	19.12	8.54
Function Points	20	120006	4809.83	153	14102	6000.3

4.2. Steps in The Estimation of Software Product Points

We executed our proposed approach "SPPSS" using the following steps as outlined in a previous research work by Abrahao *et al*. [1] and Aris and Salim [2].

- Every software services company maintain A Project Database (PDS) that would comprise of the dataset such as costs, function points, efforts, and lines of code and other hardware/software requirements/configuration details of their projects. Such a PDS is viewed as a repository of completed projects of a Company. Mostly, this database would be an in-house product of the respective company to document the details of their previously completed IT/IS projects.
- Though software startups, in general, are less interested in project documentation, a few of them that are more than 5 years old have started maintaining such databases to document their projects in the recent past. PDS is expected to provide the following details: actual efforts the company had used in a software project, lines of code, function points, project planning/execution and project scheduling, software metrics used to measure quality (for e.g., defects).
- Dataset that we collected from the Tenic's PDS comprises of the data pertaining to 20 of their completed projects' Actual Efforts (AE) and FP.
- Using the Equation (1), the MP for 20 software projects (P_i) of the startup company 'Tenic' was estimated with the support of the technical team members involved in software development at Tenic. This was followed by the calculation of the PP.
- The Estimation of Efforts (EF) for 20 software projects (P_i) of Tenic using the newly estimated software product points were done using the method namely COCOMO II for effort estimation [4].
- The function points measurement for these 20 projects (P_i) was done using the Function Point Analysis (FPA).
- As observed by software engineering researchers namely Ceke and Milasinovic [5] and Daneva [8], we performed the correlation analysis for the variables, the FP and the AE; the SPP and the EE; the SPP and the AE so as to ascertain the accuracy of the results obtained using our proposed approach "SPPSS".

5. Results

We now present our obtained results with respect to:

1. Correlation analysis between the FP and the AE of 20 software projects (P_i).
2. Correlation analysis between the estimated SPP and the Estimated Efforts (EE) of P_i.
3. Correlation analysis between the estimated SPP and the AE of P_i.

In this research work, as a first step, we documented the function points and the actual efforts used for those 20 software projects (P_i) of the software startup company 'Tenic' as shown in Table 2. Next, we estimated the software product points as per the procedure described in the Figure 1. Thereafter, the efforts required for those 20 projects (P_i) was measured using software product points. The computed values were recorded in a separate Table 3. This table shows the software product points and the efforts (both actual and estimated) for the chosen 20 software projects (P_i).

Table 2. Function points and efforts of 20 software projects (P_i) of tenic.

Project ID (P_i)	FP	AE	Project ID (P_i)	FP	AE
P_1	153	1.99	P_{11}	11002	18.77
P_2	3567	7.69	P_{12}	7543	13.88
P_3	167	10.99	P_{13}	6900	5.21
P_4	1201	11.32	P_{14}	14002	6.99
P_5	9989	19.12	P_{15}	1775	1.16
P_6	1799	18.87	P_{16}	9799	3.88
P_7	237	3.91	P_{17}	11012	4.11
P_8	1080	5.23	P_{18}	6999	6.17
P_9	1675	12.89	P_{19}	7105	7.99
P_{10}	9899	7.12	P_{20}	14102	3.54

Table 3. Software product points and efforts for 20 software projects (Pi) of Tenic.

Project ID (P_i)	SPP	EE	AE	Project ID (P_i)	SPP	EE	AE
P_1	169.8	2.91	1.99	P_{11}	1569	24.01	18.77
P_2	601.23	8.33	7.69	P_{12}	1499	15.44	13.88
P_3	708.21	11.99	10.99	P_{13}	423	5.21	5.21
P_4	854.01	15.42	11.32	P_{14}	1199	8.9	6.99
P_5	1921	20.09	19.12	P_{15}	214	2.26	1.16
P_6	1586	21.05	18.87	P_{16}	314	4.25	3.88
P_7	309	4.77	3.91	P_{17}	414	4.96	4.11
P_8	441.03	6.21	5.23	P_{18}	514	5.21	6.17
P_9	1456.01	14.09	12.89	P_{19}	411	9.33	7.99
P_{10}	523	6.99	7.12	P_{20}	198.02	3.12	3.54

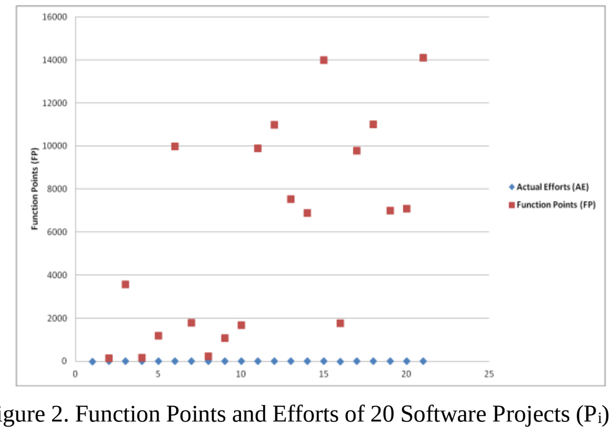

Figure 2. Function Points and Efforts of 20 Software Projects (P_i) of Tenic.

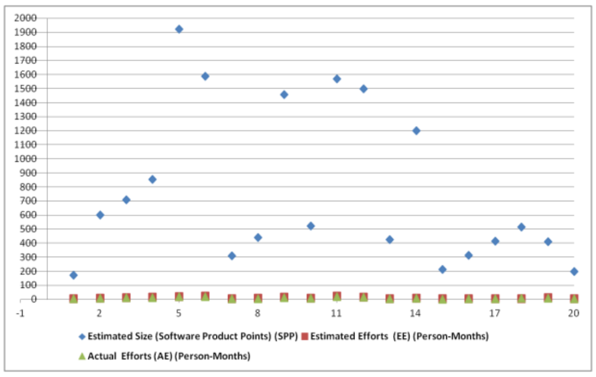

Figure 3. Software product points and efforts for 20 software projects (P_i) of Tenic.

As we have previously pointed out in section 4.2, we did the correlation analysis between the variables- the FP and the AE, the SPP and the EE, and the SPP and the AE using Karl Pearson's coefficient of correlation (R) for the 20 software projects (P_i) chosen in this study. The coefficient of correlation (R) for the FP and the AE is 0.10. It was 0.93 for the correlation analysis between the SPP and the estimated efforts. The coefficient of correlation (R) for the software product points and the actual efforts was observed as 0.94. From this analysis and the obtained results, we observe that the function points and the actual efforts used in these 20 software projects (P_i) of Tenic are weakly related. However, on the other hand, it is interesting to note that the estimated software product points and the estimated efforts are strongly related. In this line of observation, we also find that the software product points and the actual efforts used in these projects (P_i) are also strongly related as observed from Figures 2 and 3.

6. Contributions to Research and Practice

Our results indicates that the function points based software size measurement is found to be far away from the actual efforts spent for the 20 software projects (P_i) of startups. On the other hand, we also find that the estimated software product points using our proposed approach "SPPSS", were closely related to the actual efforts that was utilised in these software projects (P_i). Moreover, the estimated efforts are found to be in line with the estimated software product points for the projects (P_i). This is another significant outcome of the application of the proposed approach "SPPSS". The proposed alternate approach "SPPSS" goes beyond the existing software engineering methods for software size measurement in numerous ways. In this approach "SPPSS", we did the software size measurement from the perspective of market in which the startup is operating upon and the user stories which should bring out the specific customer's problems/requirements for the software product and, in that we leverage our acquired knowledge on efforts estimation process for software projects. Such a dimension of software size measurement for startup helps us learn the following: first, it gives a unique focus to market requirements at micro level; second it enables us to consider the constraints and the challenges that encompass the market and the product requirements during the process of software size measurement.; third, the elicitation of the varied number of tasks/ processes that would be required to those challenges in implementing certain requirements. This research work has got some significant implications for software startup companies. They are: the approach is grounded not just on the product requirements but also on other market requirements that may add up value for the product, optimal utilization of resources thereby increased productivity, and finally accuracy in effort estimation to the extent possible. The proposed approach "SPPSS" uses the SPP for software size measurement and the SPP is computed based on the market points and the product points. Such alternate mechanism would enable the project managers of startups to better understand how challenging the market requirements and the product requirements are, project efforts required to overcome those challenges and the costs it amounts to the project as a whole for building the software product. This should be of great help to the startups to precisely estimate the project costs also at the initial stage of negotiation with their customers so that in future, during the progress of the project they would never overrun the estimated project costs.

6.1. Limitations and Scope for Future Research

We examined the following limitations that may impact the obtained results.

- We have considered 20 software projects for validating our proposed approach "SPPSS". If we could consider more number of projects from startups then we would have executed our proposed approach with a much larger dataset.
- Wieringa [19] observes that a newly proposed approach or method shall be validated for its correctness and feasibility in a controlled experimental setting with minimal dataset and learn lessons from it. Thus, in this line, we have also considered sizeable projects for our validation process.
- In future, we plan to involve a few more startup companies (not just one company) and consider dataset of at least 10 to 15 software projects from each one of them thereby totalling a number of 35 to 45 projects.
- In the present research work, all the 20 software projects that we have considered for the application of the "SPPSS" are from the same startup company "Tenic". In future, as suggested previously we would like to involve a few more startup companies so as to ascertain as whether the obtained results using the

"SPPSS" is vendor independent.

7. Conclusions

In this research work, we have presented an alternate approach to measure the software size at the initial stage of the software project itself by the software startup. We have also demonstrated the application of the approach with a real dataset of 20 software projects obtained from a software startup. It is also evident that the utilization of this approach would demand a very limited resources and time for the startup.

References

[1] Abrahao S., Gomez J., and Insfran E., "Validating A Size Measure for Effort Estimation in Model-Driven Web Development," *Information Sciences*, vol. 180, no. 20, pp. 3932-3954, 2010.

[2] Aris H. and Salim S., "State of Component Models Usage: Justifying the Need for a Component Model Selection Framework," *The International Arab Journal of Information Technology*, vol. 8, no. 3, pp. 310-317, 2011.

[3] Blank S., "Why the Lean Start-Up Changes Everything," *Harvard Business Review*, vol. 91, no. 5, pp. 63-72, 2013.

[4] Boehm B., Abts C., and Chulani S., "Software Development Cost Estimation Approaches-A Survey," *Annals of Software Engineering*, vol. 10, no. 1, pp. 177-205, 2000.

[5] Ceke D. and Milasinovic B., "Early Effort Estimation in Web Application Development," *Journal of Systems and Software*, vol. 103, pp. 219-237, 2015.

[6] Centobelli P., Cerchione R., and Esposito E., "Knowledge Management in Startups: Systematic Literature Review and Future Research Agenda," *Sustainability*, vol. 9, no. 3, pp. 361, 2017.

[7] Coleman G. and Connor R., "An Investigation Into Software Development Process Formation in Software Start-Ups," *Journal of Enterprise Information Management*, vol. 21, no. 6, pp. 633-648, 2008.

[8] Daneva M., "Balancing Uncertainty of Context in ERP Project Estimation: an Approach and A Case Study," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 22, no. 5, pp. 329-357, 2010.

[9] Giardino C., Paternoster N., Unterkalmsteiner M., Gorschek T., and Abrahamsson P., "Software Development in Startup Companies: The Greenfield Startup Model," *IEEE Transactions on Software Engineering*, vol. 42, no. 6, pp. 585-604, 2016.

[10] Giardino C., Bajwa S., Abrahamsson p., and Wang X., "Key Challenges in Early-Stage Software Startups," *in Proceedings of International Conference on Agile Software Development*, pp. 52-63, 2015.

[11] Gomes J., Montenegro J., Canto dos Santos J., Barbosa J., and Costa C., "A Strategy Using Continuous Simulation to Mitigate Effort Estimation Risks in Software Projects," *IEEE Latin America Transactions*, vol. 17, no. 8, pp. 1390-1398, 2019.

[12] Klotins E., Unterkalmsteiner M., andGorschek T., "Software Engineering in Start-Up Companies: An Analysis of 88 Experience Reports," *Empirical Software Engineering*, vol. 24, no. 1, pp. 8-102, 2019.

[13] Klotins E., Unterkalmsteiner M., and Gorschek T., "Software-Intensive Product Engineering in Start-Ups," *IEEE Software*, vol. 35, no. 4, pp. 44-52, 2018.

[14] Melegati J. and Goldman A., "Requirements Engineering in Software Startups: A Grounded Theory Approach," *in Proceedings of International Conference on Engineering, Technology and Innovation/IEEE lnternational Technology Management Conference*, Trondheim, pp. 1-7, 2016.

[15] Morgenshtern O., Raz T., and Dvir D., "Factors Affecting Duration and Effort Estimation Errors in Software Development Projects," *Information and Software Technology*, vol. 49, no. 8, pp. 827-837, 2007.

[16] Paternoster N., Giardino C., Unterkalmsteiner M., Gorschek T., and Abrahamsson P., "Software Development in Startup Companies: A Systematic Mapping Study," *Information and Software Technology*, vol. 56, no. 1, pp. 1200-1218, 2014.

[17] Santisteban J. and Mauricio D., "Systematic Literature Review of Critical Success Factors of Information technology Startups," *Academy of Entrepreneurship Journal*, vol. 23, no. 2, pp.1-23, 2017.

[18] Unterkalmsteiner M., Wang X., Abrahamsson P., et al., "Software Startups-A Research Agenda," *E-Informatica Software Engineering Journal*, vol. 10, no. 1, pp. 1-28, 2016.

[19] Wieringa R., "Empirical Research Methods for Technology Validation: Scaling Up to Practice," *Journal of Systems and Software*, vol. 95, pp. 19-31, 2014.

[20] Wohlin C., Runeson P., Host M., Ohlsson M., Regnell B., and Wesslen A., *Experimentation in Software Engineering*, Springer Science and Business Media, 2012.

[21] Yin R., *Case Study Research: Design and Methods*, Sage Publications, 2013.

[22] Yucalar F., Kilinc D., Borandag E., and Ozcift A., "Regression Analysis Based Software Effort Estimation Method," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 5, pp. 807-826, 2016.

Chandrasekaran Sridharan obtained his B.E. degree in Electronics and Communication Engineering from PSG College of Technology, University of Madras, in the year 1986. He obtained his Master of Engineering degree from Madras Institute of Technology, Anna University, Chennai in 1988. He is currently doing Ph.D under Anna University, Chennai. He has been serving as a faculty in Thiagarajar College of Engineering, Madurai for the past thirty three years. Presently, he is an Associate Professor in Computer Science and Engineering at Thiagarajar College of Engineering, Madurai. He has published several papers in reputed conferences and journals. He was the Principal Investigator for his institute in the Collaborative Research Project on Smart and Secure Environment sponsored by the Government of India. His area of interest include Computer Networks, Information Security and Software Engineering

Sudhaman Parthasarathy is working as a Professor of Data Science in the Dept of Applied Mathematics and Computational Science, Thiagarajar College of Engineering, Madurai, India. He obtained his Ph.D. in Computer Applications from Anna University, Chennai in 2009. A habitual rank holder, he has been teaching at the post-graduate level for the past 15 years. He has published research papers in peer reviewed conferences and International Journals such as Journal of Systems and Software, Computers in Industry, Software Quality Journal, International Journal of Project Management and Business Process Management Journal. He has authored several chapters in the refereed edited books of IGI, USA and Springer, London. He is the Editor-in-Chief for the Edited Book ‘‘Enterprise Information Systems and Implementing IT Infrastructures: Challenges and Issues’’ which was published in May 2010. His current research interests include enterprise information systems, ERP and software engineering.