THE LEAN SIX SIGMA PRACTICES IN SOFTWARE DEVELOPMENT ORGANIZATION USING INTERPRETIVE STRUCTURAL MODELLING APPROACH

Dr. Chandrakanth G Pujari[1], Kavyashree N[2], Dr.Supriya M C[3]
[1] Professor, MCA Dept, Dr.AIT, Bangalore.
[2] Research Scholor, SSIT Tumkur, Asst.Professor, MCA Dept. Dr.AIT, Bangalore,
[3] Associate Professor, Dept. of MCA, SSIT, Tumkur.

Abstract
The important functions of lean in software development practices has enhanced with a maximum competition in global software worldwide business. A software customer would like to purchase a software product at the minimum possible cost without compromising software reliability and software quality. To defeat this serious problem, the function of lean in software development practices has been used. Lean Six Sigma is ability to do something for waste reducing without any compromising in software reliability, software productivity, and software quality. For achieving this intention, thirteen software practices of lean in software development practices have been selected through literature survey, six sigma Champions advice, from academicians, and organizational experts. To select the annoyed impact of lean in software development practices, the framework modelling approach have been used to discover interrelation between these lean six sigma software development process practices utilised. By using "cross-impact matrix product applied to classification" is applied for several areas with interpretation self structure modelling. Output of this examinations are highest profitable towards the execution in software organizations getting lean six sigma methodology.

Keywords: Interpretive Structural Modelling Approach, Lean Six Sigma, Software practices and process, Software Industry

1. Introduction

The lean software development processes has been advised by Krafcik in the year 1988, and he described a software development process that utilised minimum to meet software project development production. Several researchers said various definitions of lean six sigma software development processes is very good direction, a group of principles, a group of tools, various methods, a software development project, a software rules, a software theme, a software project development project, a software project development system, a software program, a software model is introduced by Bhamu and Singh Sangwan in 2014. To reach the software customer requirements in software industries need to act and move fast to survive with competitors. Lean six sigma is a methodology to minimise software development duration and production expenditure to enhance software product performance, software

reliability, and software quality. This network is constructed a number of links. This is within the software company and government sectors. Lean Six Sigma is a multidimensional approach for software production with minimum cost of waste, very good maintenance hardware equipment, efficient skilled empowered software developer and very good established software quality. Software cost, software quality, software flexibility, and quick response this chain impact on competitive performance.

Lean six sigma software development is attention on removal of wastes and enhancing software customer satisfaction. This lean six sigma methodology is a important for waste minimization inside a software development system with maximize in software production. Waste can be anything like people, material, software developer etc. which does not lead to value addition of the software production. In this article, lean six sigma software practices are recognized by Champions advice and literature survey and a framework modelling method is used to verify software process. Important objective of this research are as follows:

- To select and study the lean six sigma practice
- To construct interpretive self structured modelling using lean six sigma software development practices.
- To advice research insights using "cross-impact matrix product applied to classification" approach.

2. Literature Survey

The important concept of lean in software development practice is implemented by different organizations. Lean six sigma principles gave the term "lean six sigma software production".

Lean six sigma methodology is implemented in Indian companies. Implementation of lean six sigma leads more profitable for the companies. In the year of 2018 Rishi et al. executed the lean methodology in the software project development practices to improve its software productivity. Another best technique called Kaizen, is used to maximize the software productivity by minimizing the lean duration. Khalili suggested in the year 2017 that software organizations should follow continuous improvement for long time prospective.

3. Reorganization of lean software development practices

In the following table1, I have been listed various lean six sigma software development and their software practices as follows:

Table I. Lean Software Development and their Software practices

Lean Six Sigma Practices	Properties
a. Bench Marking	Is a lean methods, comparing with any one software company with another software company
b. Bottleneck Removal	Minimize the capacity of the software production system
c. Software development projects Continuous Improvement	Is vital to meet the software company targets, incremental enhancement of software development projects, software process, software services, maximum duration with the objective of software waste minimization to enhance the software performance and minimize the software failures.
d. Software development Cross functioning workforce	A software project development cross functioning workforce group is a set of software developers with several function expertises towards a same achievement. It may consists of software developers, from finance section, marketing division, software operation and HR department
e. Software development Time minimization	Software development time is the time taken to finish one software project product or is the total time taken before leaving the software product from software development division
f. Just In Time (JIT)	The objective of this method is to maximize the software projects development by minimizing the inventory. By using this method to minimize software development in process inventories and releasing the software projects when it is needed
g. Lot Size reduction	Is the quantity of software projects developed in only one software project development run, for software projects in maximum lot size, maximum capital and software development process needed. Highest software projects size also maximize needed and software projects managing.
h. Software Projects Maintenance Optimization	In software company, project management expenditure is significant part of the entire expenditure of software project. The software management section expenditure is 15% to 70% expenditure of the entire software development projects expenditure.
i. Software development planning and scheduling strategy	The main objective of Planning and Scheduling Strategy is to release their software production and supply to software customer within the period and minimizes the software development lead duration to meet software customer demand.
j. Software Development Process Capability	Combination of software, software developers, System, software tools, and evaluation to produce software project that will join the

Measurement	software plan needed and software client expectation.
k. Kanaban	Is a scheduling system in lean six sigma software development
l. Software Reengineering Production Process	Is Software development re-thinking and software development re-designing software development projects to achieve the improvement of expences, quality, services and speed.
m. Software Development TQM	Is a methodology for enchanting the software improvement of the software projects, software process and software services

4. Technique used

In literature survey, various types of tools and techniques have been used such as principle component analysis, factor analysis, and multiple component analysis. In this research interpretive self structure modelling approach has been used to analyse the lean six sigma methodology processes. This is a important and broadly used decision making philosophy in literature survey for the varieties of component analysis war field was proposed in 1974. Strong methodology for interpreting self structure modelling is as follows:

- Selecting the parameters related to the software development problem.
- Setup circumstances surrounding an event relationship between parameters w,r,t which pairs of parameters would be tested.
- Construct framework of self interaction matrix of parameters.
- Construct a matrix of reachability using structural self interaction matrix.
- Divide the reachability matrix into several phases.
- Convert reachability matrix into conical form.
- Hierarchy of parameters is formed.

5. Analysis of Lean six sigma practices using Interpretive structural modelling

The development of interpretive structural modelling are explained below steps

Step 1 SSIM (Structural self-intersection matrix)

The circumstances surrounding an event of relations in between the software elements are selected by the champions of academic and software organization. Following notations are used to indicate the relationships, having ith row and jth column

- X->Xij: Software factor i = 1 to n and j = 1 to m, row i lead to column j
- Y->Yij: Software element i = 1 to n and j = 1 to m, column j will lead to row j
- Z->Zij: Software factor i = 1 to n and j = 1 to m, row i and column j lead to each other

Table II SSIM (Structural self-intersection matrix)

Software practices	M	l	k	j	i	h	g	f	e	d	c	b
a	U	Z	U	U	Y	U	U	U	X	U	Y	U
b	X	X	U	X	U	U	U	U	U	U	X	
c	Y	Z	Y	U	Y	Y	U	Y	X	Y		
d	U	X	U	U	X	U	U	U	U			
e	Y	Y	Y	X	Y	Y	U	U				
f	X	X	U	U	Y	Y	X					
g	U	U	U	U	U	U						
h	U	U	U	U	U							
i	U	Z	U	U								
j	U	U	U									
k	U	U										
l	Y											

- U->Uij: Software factor i = 1 to n and j = 1 to m, row i and column j having no relation between

the software factors

Structural self-interaction matrix has been developed on the basis of the circumstances surrounding an event relation among the important software factors.

Step 2 RM (Reachability Matrix)

Convert the structural self-intersection matrix into binary values T (true) and F (false). This procedure is initially named as reachability matrix

Table III Initial Reachability Matrix

Software Practices	a	b	c	d	e	f	g	h	i	j	k	l	m
a	T	F	F	F	T	F	F	F	F	F	F	T	F
b	F	T	T	F	F	F	F	F	F	T	F	T	T
c	T	F	T	F	T	F	F	F	F	F	F	T	F
d	F	F	T	T	F	F	F	F	T	F	F	T	F
e	F	F	F	F	T	F	F	F	F	T	F	F	F
f	F	F	T	F	F	T	T	F	F	F	F	T	T
g	F	F	F	F	F	T	T	F	F	F	F	F	F
h	F	F	T	F	T	T	F	T	F	F	F	F	F
i	T	F	T	F	T	T	F	F	T	F	F	T	F
j	F	F	F	F	F	F	F	F	F	T	F	F	F
k	F	F	T	F	T	F	F	F	F	F	T	F	F
l	T	F	T	F	T	F	F	F	F	F	F	T	T
m	F	F	T	F	T	F	F	F	F	F	F	T	T

Conversion of SSIM into RM by replacing X, Y, Z, and U into T and F. The following procedure can be followed:

(1) If the ith row and jth column entry in the structural self-intersection matrix is X then put it into the (ith row, jth column) in the reachability matrix as T and (jth column, ith row) entry as F

(2) If the (ith row, jth column) entry in the structural self-intersection matrix is Y then put it into the (ith row, jth column) entry in the reachability matrix as F and (jth column, ith row) entry as T

(3) If the (ith row, jth column) entry in the structural self-intersection matrix is Z then put it into the (ith row, jth column) entry in the reachability matrix as T and (jth column, ith row) entry as T

(4) If the (ith row, jth column) entry in the structural self-intersection matrix is U then put it into the (ith row, jth column) entry in the reachability matrix as F and (jth column, ith row) entry as F

Table III is called as initial reachability matrix which is obtained by the pictorial intersection matrix, now last reachability matrix is using law of transitivity relation applied in initial reachability matrix as in Table II. law of transitivity relation between three factors if relationship holds between the 1st and 2nd factors, 2nd and 3rd factors, then relationship must holds between the 1st and 3rd factors. Now Table IV is obtained by using transitivity axioms denoted by $T^{\#}$ for example T is related to e and software practice e is related to j then project T is related to project j indicated with $T^{\#}$ in Table IV

Table IV Final reachability matrix

Software Practice	a	b	c	d	e	f	g	h	i	j	k	l	m
a	T	F	$T^{\#}$	F	T	F	F	F	F	$T^{\#}$	F	T	$T^{\#}$
b	$T^{\#}$	T	T	F	$T^{\#}$	F	F	F	F	T	F	T	T
c	T	F	T	F	T	F	F	F	F	$T^{\#}$	F	T	$T^{\#}$
d	$T^{\#}$	F	T	T	$T^{\#}$	$T^{\#}$	F	F	T	F	F	T	$T^{\#}$
e	F	F	F	F	T	F	F	F	F	T	F	F	F
f	$T^{\#}$	F	T	F	F	T	T	F	F	F	F	T	T
g	F	F	$T^{\#}$	F	F	T	T	F	F	F	F	F	$T^{\#}$
h	$T^{\#}$	F	T	F	T	T	$T^{\#}$	T	F	$T^{\#}$	F	F	$T^{\#}$
i	T	F	T	F	T	T	$T^{\#}$	F	T	$T^{\#}$	F	T	$T^{\#}$
j	F	F	F	F	F	F	F	F	F	T	F	F	F
k	$T^{\#}$	F	T	F	T	F	F	F	F	$T^{\#}$	T	$T^{\#}$	F
l	T	F	T	F	T	F	F	F	F	$T^{\#}$	F	T	T
m	$T^{\#}$	F	T	F	T	F	F	F	F	$T^{\#}$	F	T	T

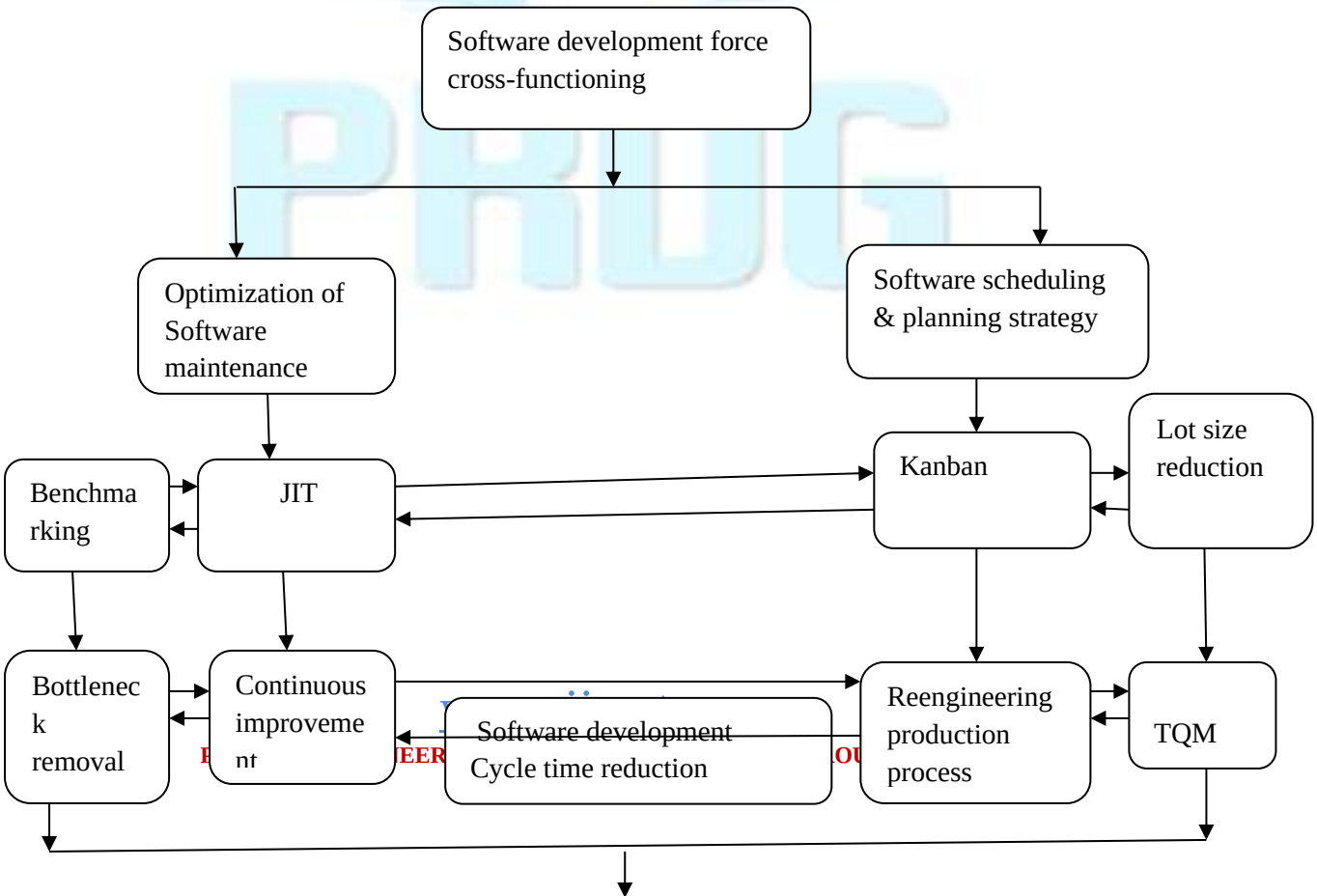
From reachability matrix Table III group of reachability (GR) and group of antecedent (GA) are reached. After computing (GR) and (GA) then group of intersection (GI) of all these groups are derived for the measures.

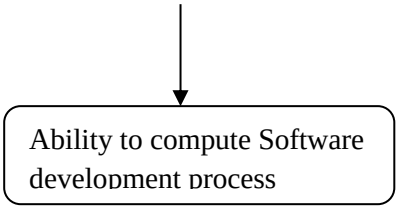
Table V 1st Iteration

Software practices	GR	GA	GI	Phase
A	a, c, e, j, l, m	a, b, c, d, f, h, i, k, l, m	a, c, l, m	
B	a, b, c, e, j, l, m	b	b	
C	a, c, e, j, l, m	a, b, c, d, f, g, h, i, k, l, m	a, c, l, m	
D	a, c, d, e, f, i, l, m	d	d	
E	e, j	a, b, c, d, e, h, i, k, l, m	e	
F	a, c, f, g, l, m	d, f, g, h, i	f, g	
G	c, f, g, l, m	f, g, h, i	f, g, m	
H	a, c, e, f, g, h, j, m	h	h	
I	a, c, e, f, g, i, j, l, m	d, i	i	
J	j	a, b, c, e, h, i, j, k, l, m	j	
K	a, c, e, j, k, l	k	k	
L	a, c, e, j, l, m	a, b, c, d, f, i, k, l, m	a, c, l, m	**1**
M	a, c, e, j, l, m	a, b, c, d, f, g, h, i, l, m	a, c, l, m	

Those measures for which (GR) and (GI) has the same elements are placed at the top level of interpretive structural modelling hierarchy. Top level computation is placed at the top digraph and so on. From table V.

Figure 1 Interpretive Structural Modelling of Lean Six Sigma Software Development Practices





In this figure demonstrates that to concentrate on software development force cross-functioning, introduced in an end of the interpretive structural modelling it affect on the software maintenance maximization and designing and scheduling plan. Dependent on each software project is spotted to enhance the software development practice computation.

Table VI 2nd Iteration

Software practice	GR	GA	GI	Phase
A	a, c, e, l, m	a, b, c, d, f, h, i, k, l, m	a, c, l, m	
B	a, b, c, e, l, m	b	b	
C	a, c , e, l, m	a, b, c, d, f, g, h, k, l, m	a, c, l, m	
D	a, c, d, e, f, i, l, m	d	d	
E	e	a, b, c, d, e, h, i, k, l, m	e	2
F	a, c, f, g, l, m	d, f, g, h, i	f, g	
G	c, f, g, l, m	f, g, h, i	f, g, m	
H	a, c, e, f, g, h, m	h	h	
I	a, c, e, f, g, i, l, m	d, i	i	
K	a, c, e, k, l	k	k	
L	a, c, e, l, m	a, b, c, d, f, i, k, l, m	a, c, l, m	
M	a, c, e, l, m	a, b, c, d, f, g, h, i, l, m	a, c, l, m	

Table VII 3rd Iteration

Software Practice	GR	GA	GI	Phase
A	a, c, l, m	a, b, c, d, f, h, i, k, l, m	a, c, l, m	3
B	a, b, c, l, m	b	b	
C	a, c, l, m	a, b, c, d, f, g, h, i, k, l, m	a, c, l, m	3
D	a, c, d, f, i, l, m	d	d	
F	a, c, f, g, l, m	d, f, g, h, i	f, g	
G	c, f, g, l, m	f, g, h, i	f, g, m	
H	a, c, f, g, h, m	h	h	
I	a, c, f, g, i, l, m	d, i	i	
K	a, c, k, l	k	k	
L	a, c, l, m	a, b, c, d, f, i, k, l, m	a, c, l, m	3
M	a, c, l, m	a, b, c, d, f, g, h, i, l, m	a, c, l, m	3

Table VIII 4th iteration

Software Practice	GR	GA	GI	Phase
B	b	b	b	4
D	d, f, i	d	d	
F	f, g	d, f, g, h, i	f, g	4
G	f, g	f, g, h, i	f, g	4

H	f, g, h	h	h	
I	f, g, i	d, i	i	
K	k	k	k	4

Table IX 5th iteration

Software Process	GR	GA	GI	Phase
D	d, i	d	d	
H	h	h	h	5
I	i	d, i	i	5

Table X 6th iteration

Software Practice	GR	GA	GI	Phase
D	d	d	d	6

Table XI 7th iteration

Software Practice	j	e	a	c	l	m	b	f	g	k	h	i	d	Power driving
J	T	F	F	F	T	F	F	F	F	F	F	F	F	2
E	T	T	F	F	F	F	F	F	F	F	F	F	F	2
A	T	T	T	T	T	T	F	F	F	F	F	F	F	6
C	T	T	T	T	T	T	F	F	F	F	F	F	F	6
L	T	T	T	T	T	T	F	F	F	F	F	F	F	6
M	T	T	T	T	T	T	F	F	F	F	F	F	F	6
B	T	T	T	T	T	T	T	F	F	F	F	F	F	7
F	F	F	T	T	T	T	F	T	T	F	F	F	F	6
G	F	F	F	T	F	T	F	T	T	F	F	F	F	4
K	T	T	T	T	T	F	F	F	F	T	F	F	F	6
H	T	T	T	T	F	T	F	T	T	F	T	F	F	8
I	T	T	T	T	T	T	F	T	T	F	F	T	F	9
D	T	T	T	T	T	T	F	T	F	F	F	F	T	8
Power dependent	11	10	10	11	10	10	1	5	4	1	1	2	1	

matrix multiplication of cross-impact used to classification analysis in different software fields, with interpretive structural modelling. This classification method is used to analyse the power dependent r and power driving of software critical elements. In this software development projects are classified into 4 groups, cluster independent, groups dependent, groups linkage, and groups driven.

6. Analysis of Classification

This analysis is used in different software fields with interpretive structural modelling, and also taken to explain the power dependent and power driving of software factors. In software development processes are classified into 4 clusters as explained below:

I-Cluster is having autonomous parameters; these software elements have weak in power power dependence. Here, cluster have 3 software process elements, such as Lot-Size minimization, Just-in-time, and Kanban.

II-Cluster is having dependent parameters; these software elements very weak in power drive but strong power dependence. In this group 6 software process elements, such as Bottle neck removal, Continuous improvement, Cycle time minimization,

III-Cluster is having linkage parameters; here software process elements have very strong power drive as strong power dependence. In this groups no factors available.

IV-Cluster having running parameters; here software process elements have very good power drive but minimum power dependence. group having 4 factors such as benchmarking, cross-functioning, workforce, and software maintenance maximization.

Table XII Lean Six Sigma Software Practice of Cluster Measures

Power of driving	**m**													
	l													
	k													
	j													
	i		**i**	**4**						**3**				
	h	**d, h**												
	g	**b**												
	f	**k**					**f**				**a, l, m, c**			
	e													
	d				**g**									
	c			**1**						**2**				
	b										**e, j**			
	a													
		a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**
Power of Dependent														

7. Summary and Conclusion

The important goal of this article is to recognize and analyse the lean six sigma software development practice and express indirectly suitable software practices to execute in Software Company. Thirteen lean six sigma practices has been recognized form by champion's suggestions from software companies and academicians. In this article, interpretive structural modelling approach based on relationship has been analysed to determine the relationship among lean six sigma software development practices. According to matrix multiplication of cross-impact classification determines, parameters in the cluster autonomous, that means power of drive is weak and power of dependent is also weak. Here we have three software practice factors such as kanban, reduction Lot-Size, Just-In-Time. Here group of cluster some of them are weak power dependent and weak power driving. These are not influence on the software development system. Cluster dependent having six software practice factors such as software development cycle- time minimization, TQM, software development bottleneck removal, software development continuous improvement, software project development capability measurement process, production process re-engineering. In this group of cluster some of them are weak power of dependence, very good power of driving. The 3rd group of group is linkage cluster having no parameter, in this cluster having very good dependent and very good driving power. Next group is driving group having 4

parameters such as software maintenance optimization, benchmarking, planning and scheduling strategy, and cross-functioning workforce, in this group of cluster having strong driving power and weak dependent. They are considered as key software practice for powerful execution of lean six sigma software development in software organization. Lean six sigma software development uses the software companies in dropping of various hidden waste. After reorganization of waste, disciplined minimizing and removing practices can be produced to make software company waste free. Finally study concludes that designing and planning strategy, cross-functioning workforce, benchmarking, and software maintenance optimization are the four software practices are identified as maximum software driving practices and minimum dependencies. Catching up these software practices to excel in software development work is highly recommended.

References

[1] Chandrakanth G Pujari and Dr. Seetharam K.," **Top Priority of Software Success Factor for Six Sigma Execution by a Fuzzy Hierarchical Process",** International Journal of Multimedia and Ubiquitous Engineering 9 (11), 171-180, 2014-15

[2] Chandrakanth G Pujari and Dr. Seetharam K.,"Evaluation for Defective Density in All the Right Places", Indian journal of engineering, 2014, 11(26), 30-37

[3] Chandrakanth G Pujari and Dr. Seetharam K."Ranking of Tools use, software logical complexity, Requirement volatility, Quality requirements, Efficiency requirements in software development", **2009 IEEE International Advance Computing Conference,**
DOI: 10.1109/IADCC.2009.4809258,Date Added to IEEE *Xplore*: 31 March 2009

[3] Chandrakanth G Pujari, "Software Information Flexibility for Lean Six sigma Software Development Using Multiple Regression Analysis", Indian Journal of Engineering, 2017, 14(36), 95-107

[4]ChandrakanthGPujari,Dr.SeetharamK,**"Investigating the Effects of Factors on Software Development",**International Journal of Computer Applications, vol. 1, issue 6, pp. 56-65, February 2010, 10.5120/142-261

[5] Chandrakanth G Pujari and Dr. SeetharamK.,**Article: "Estimation of Growth ParametersforaSoftwareDevelopment". *International Journal of Computer Applications* 35(12):38-42, December 2011.**

[6] Chandrakanth G Pujari, Kavyashree N, Dr.Supriya M C "**Enhancement of Indian Software Quality Management Using Multi Criteria Objects and Six Sigma Methodology**", International Journal on Future Revolution in Computer Science & Communication Engineering ISSN: 2454-4248 Volume: 4 Issue: 4 806 – 81 2019

[7] Chandrakanth G Pujari, Dr. SeetharamK ,"**Detection and valuation of major error trends of software projects using paret principle and fuzzy model**", National journal on advances in computing & Management, vo 3 no. 2 october 2012.

[8] Chandrakanth G Pujari, Kavyashree N Dr.Supriya M C, "**Development of Methodology for Software Small an Medium Scale Industries in the Selection o Suitable Lean Six Sigma Tools**", JASC Journal of Applied Science and Computation Volume VI, Issue II, February/2019 ISSN NO

1076-5131

[9] Chandrakanth G Pujari and Dr. Seetharan K., "An Evaluation of Effectiveness of the **Software Projects Developed Through Six Sigma Methodology**", American Journal of Mathematical and Management Sciences, **34(1) · January 2015-16**

[10] Chandrakanth G Pujari and Dr. Seetharam K, "**Software Defects Identification Using Principles of Data Gathering and Pareto Analysis" ,**DOI**: 10.5176/2251-2217_SEA12.36, 2012**

[11] Chandrakanth G Pujari and Dr. Seetharam K," **Software Defects Identification Using Principles of Data Gathering and Pareto Analysis",** DOI**: 10.5176/2251-2217_SEA12.36 2012**

[12] Chandrakanth G Pujari , "Modeling Software Project Defects With Fuzzy Logic Maps**", International Journal on Future Revolution in Computer Science & CommunicationEngineering ISSN: 2454-4248 Volume: 4 Issue: 4 103 – 107, IJFRCSCE | April 2018Available@http://www.ijfrcsce.org**

[13] **Chandrakanth G Pujari and Dr. Seetharam K," Implementation of Multivariate Clustering Methods for SoftwareDevelopment**", https://www.bvicam.ac.in

[14] Buglione, L., Trudel, S. (2010), "**Guideline for sizing Agile projects with COSMIC**",In: Proceedings of the IWSM / MetriKon / Mensura 2010, Stuttgart, Germany

[15] Bhasin, S. (2011), "**Performance of organizations treating lean as an ideology**", Business Process Management Journal, Vol. 17 No. 6, pp. 986-1011.

[16] Bergmiller, G.G. and McCright, P.R. (2009), "**Parallel models for lean and green operations**", paper presented at Industrial Engineering Research Conference, Miami, FL.

[17] Furlan, A., Vinelli, A. and Dal Pont, G. (2011), "**Complementory and lean manufacturingbundles: an empirical analysis**", International Journal of Operations & Production Management, Vol. 31 No. 8, pp. 835-850.

[18] George, M. ed. (2010), "**The Lean Six Sigma Guide to Doing More with Less cost"**, John Wiley & Sons, Hoboken,NJ

[19] Stone, K.B. (2012), "**Four decades of lean: a systematic literature review**", International Journal of Lean Six Sigma, Vol. 3 No. 2, pp. 112-132.

[20] Vinodh, S. and Balaji, S. (2011), "**Fuzzy logic based Leanness assessment and its decisionsupport system**", InternatiofJournal of Production Research, Vol. 49, pp. 40-67.

[21] Zhang, Q., bbas, J., Zhu, X. and Shah, M. (2012), "**Critical success factors for successful Lean Six Sigma implementation in Pakistan**", Interdisciplinary Journal of Contemporary Research in Business, Vol. 4 No. 1, pp. 117-124.

[22] S.M. Kazemi, (16-18 July 2012). Six Sigma project selection by using a fuzzy multiple criteria decision making approach a case study in poly acryl corp. CIE42 proceedings, Cape Town, South Africa 2012 CIE & SAIIE.

[23] Snee, Ronald: Leading Six Sigma: A Step-by-Step Guide Based on Experience with GE and Other Six Sigma Companies, Financial Times Prentice Hall 2002

[24] Stone, K.B. (2012), "Four decades of lean: a systematic literature review", ternational Journal of Lean Six Sigma, Vol. 3 No. 2, pp. 112-132.

[25] Vinodh, S. and Balaji, S. (2011), "Fuzzy logic based Leanness assessment and its decision support system", International Journal of Production Research, Vol. 49, pp. 40-67.

[26] Furlan, A., Vinelli, A. and Dal Pont, G. (2011), "Complementory and lean manufacturing bundles:empiricalanalysis", International Journal of Operations & Production Management, Vol. 31 No. 8, pp. 835-850.

[27] Maroofi, F. and Dehghan, S. (2012), "Performing lean manufacturing system in smal and mediumenterprises",International Journal of Academic Research in Accounting, Finance and Management Sciences, Vol. 2 No. 3 pp. 156-163.
[28] Nordin, N., Deros, B.M. andAbdWahab, D. (2012), "A framework for managing change in lean manufacturingimplementation", International Journal of Services and Operations Management, Vol. 12 No. 1, pp. 101-117.
[29] Stone, K.B. (2012), "Four decades of lean: a systematic literature review", International Journal of Lean SixSigma, Vol. 3 No. 2, pp. 112-132.