International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

A Quantitative Analysis of Defect Prevention in E-commerce Projects

R. Sudarshan, S. K. Srivatsa

Abstract*: *E-commerce is a transaction of buying or selling online. It is a methodology of modern business which addresses the need of business organizations, vendors and customers to reduce cost and improve the quality of goods and services while increasing the speed of delivery [1]. In the emerging global economy, e-commerce and e-business have increasingly become a necessary component of business strategy and a strong catalyst for economic development. E-commerce promises better business for Small and Medium Enterprises (SME) and sustainable economic development for developing countries [2]. With the advent of e-commerce business, many software providers have started developing e-commerce solutions. However, the emphasis has to be given on developing defect free e-commerce portals. The concept of defect removal effectiveness and its measurement are central to software development. As per Software Engineering Institute Website defect removal is one of the top expenses in any software project and it greatly affects schedules [3]. According to Caper Jones, for improvements in productivity, cost, and schedule, it is important to use better defect prevention and removal techniques [4]. This quantitative analysis is done on five e-commerce projects. The lessons learnt from one project are fed into the development process of the subsequent projects that helped in preventing the similar defects. This facilitated in reduction in the count of User Acceptance Test defects, in the projects developed subsequently.

Index Terms*: *defect origin and detection, defect removal efficiency, defect metric analysis, e-commerce Web portal..

I. INTRODUCTION

An important strategy of any Information Technology organization is to deliver defect free or quality products. However, developing the quality product comes with a price tag. Joseph M Juran, developed the "Cost of Quality" concept – a measure that takes into account all the costs associated with good and poor quality [5]. According to him the cost of quality represents the costs to achieve good quality as well as the costs incurred due to poor quality. Cost of poor quality attributes to internal failure costs and external failure costs. *Internal failure costs* - the cost incurred due to the defects, before shipping the products to production or live environment. This is preproduction cost. *External failure costs* - the cost incurred due to the defects, after shipping the products to production or live environment. This is post production cost. Not all defects in the software development are caused by coding errors. In the software development lifecycle (SDLC), the defects could be due to:

- misunderstanding of requirements
- missing requirements
- wrong analysis and documentation of requirements
- incomplete requirements
- inability to identify non-functional requirements
- incorrect architecture or design

Since the E-Commerce portal development includes a number of integrations like Vendor Information, Buyer Information and Authentication, Inventory, Payment Gateways, Delivery Logistics etc., it becomes very important to minimize defects at the end of each phase of developmental activity.

II. SOFTWARE DEFECT ANALYSIS

This analysis is based on the study on defects that had emerged from various phases of software development like Requirements Analysis, Design, Coding (with Unit Testing), Integration Testing and User Acceptance Testing. Effective defect removal can lead to reductions in the development cycle time and good product quality [6].

A. Defect Removal Efficiency

To define defect removal effectiveness clearly, we must first understand the activities in the development process that are related to defect injections and to removals. Defects are injected into the product or intermediate deliverables of the product (e.g., design document) at various phases. It is wrong to assume that all defects of software are injected at the beginning of development. For the development phases before testing, the development activities themselves are subject to defect injection, and the reviews or inspections at end-of-phase activities are the key vehicles for defect removal. For the testing phases, the testing itself is for defect removal. When the problems found by testing are fixed incorrectly, there is another chance to inject defects [7].

Defect Removal Efficiency of any development phase is given by the formula [7]:

$$\frac{Defects\ Removed}{Defects\ Existing\ +\ Defects\ Injected} \quad (1)$$

Where Defects Removed = Defects Removed at the stage
Defects Existing = Defects Existing on stage entry
Defects Injected =

Revised Manuscript Received on 30 July 2019.
***** Correspondence Author
R. Sudarshan*, Research Scholar, Department of Computer Science and Engineering,, VELS University, Chennai 600 117, India
Dr. S. K. Srivatsa, Professor, Department of Computer Science and Engineering, Anna University, Chennai 600 025, India.

Defects Injected at the stage

The concept of defect removal effectiveness and its measurement are central to software development. Defect removal is one of the top expenses in any software projects and it greatly affects schedules. For improvements in productivity, cost, and schedule it is important to use better defect prevention and removal techniques [7].

B. Defect Origins and Defect Discovery

Table I shows the methods adopted for finding defects within a specific phase or within a short time interval from when the defects originated.

Table I: Defect origin and discovery methods adopted

Defect Origin	Defect Discovery Methods
Requirements defects	Formal requirements inspections
Design defects	Formal design inspections
Coding defects	Static analysis Formal code inspections Unit testing
Bad fixes	Re-inspection after the fix Regression Testing
Test case defects	Formal test case inspections
Integration test defects	Re-inspection after the fix Regression Testing
Document defects	Formal document inspections

Defect Data Analysis

This study has been performed on five e-commerce applications sequentially developed using ATG framework. For the sake of reference identity, they are called Project A, Project B, Project C, Project D and Project E. The size of each project was around 90 Executable Kilo Line of Code (EKLOC) and in terms of effort it was around 3000 person days. The average team experience in relevant technology is greater than 5 years for all the five projects. For each project starting from Project A, the Inspection Process of Requirements Documents, Design, Test Cases was improved based on the data analysis of previous project(s). To derive an operational definition, Stephen H. Kan [7] proposes a matrix approach by cross-classifying defect data in terms of the development phase in which the defects are found and the phases in which the defects are injected. For each defect found, its origin (the phase where it was introduced) be decided by the inspection (review) group (for review defects) or by agreement between the tester and the developer (for testing defects). Once the defect matrix is established, calculations of various effectiveness measures are straight forward.

The e-commerce project development includes several modules and a large number of integrations. Each module is constructed under four phases of development life cycle of *Rational Unified Process* (RUP) [9]. Profound uncertainty prevails at the final stages of life cycle as to which defect belongs to which phase of software development cycle. The need is to introduce a process to determine the origin of those defects and ways to eliminate them at the injected step itself.

In this work, the origin of a defect is mapped to the developmental phase to which it belongs to. Each defect is given a *weight* based on the criticality of impact on the operation of the software. Table II lists the classification of defects based on severity that is adopted in the organization.

Each project's defect data (recorded for both Inspection and Testing) is presented in the Tables III to VII. A matrix approach is adopted by cross-classifying defect data in terms of the development phase in which the defects are found and the phases in which the defects are injected. Once a development organization begins collecting software data, there is a tendency for over collection and under analysis. To transform raw data into meaningful information, and to turn information into knowledge, analysis is the key [8].

The severity of the defects based on the impact of the defect in terms of timeline, risks, debugging etc., makes some defects either truly negligible or to have a major impact. To adjust the effects of such defects, a severity factor has been introduced to adjust the defect count to be more realistic for our metrics purposes.

Table II: Classification of defect data based on criticality

Type of Criticality	*Weightage* Factor	Remarks
Show Stopper	10	These are the extremely severe defects, which have the potential to halt or already halted business system
Critical	8	These are also severe defects, which have not halted the system, but have seriously degraded the performance of some business operation
High	7	These are defects which have an impact on the functionality of some modules
Medium	5	These are defects which have an adverse effect on the general transactions
Low	2	These types of defects are the ones which are primarily related to the presentation layer (cosmetic)

Table III: Defect data of Project A

Defect Distribution by Origin For Project A						No. of Defects based on Criticality (i.e. Severity)					
<Phase>	Requirements Analysis	Design	Code with Unit Testing	Integration Testing	Total	Show Stopper	Critical	High	Medium	Low	Weighed Number of Defects
Analysis	132				132	21	11	35	11	54	706
Design	18	90			108	27	18	33	30	0	795
Code with UT	78	108	243		429	24	45	81	117	162	2076
Integration Testing	10	12	363	459	844	45	72	81	281	365	3728
UAT	231	122	212		565	93	81	56	155	180	3105
Total	469	332	818	459	2078	210	227	286	594	761	10410

Table IV: Defect data of Project B

Defect Distribution by Origin For Project B						No. of Defects based on Criticality (i.e. Severity)					
<Phase>	Requirements Analysis	Design	Code with Unit Testing	Integration Testing	Total	Show Stopper	Critical	High	Medium	Low	Weighed Number of Defects
Analysis	261				261	36	36	45	63	81	1440
Design	21	108			129	6	12	27	39	45	630
Code with UT	48	63	251		362	25	42	75	108	112	1875
Integration Testing	63	23	360	247	693	54	109	112	180	238	3572
UAT	153	18	90		261	18	31	61	49	102	1304

Table V: Defect data of Project C

Defect Distribution by Origin For Project C						No. of Defects based on Criticality (i.e. Severity)					
<Phase>	Requirements Analysis	Design	Code with Unit Testing	Integration Testing	Total	Show Stopper	Critical	High	Medium	Low	Weighed Number of Defects
Analysis	342				342	27	30	66	107	112	1731
Design	33	207			240	12	31	54	51	92	1185
Code with UT	36	45	154		235	13	21	40	71	90	1113
Integration Testing	50	18	303	190	561	37	72	87	160	205	2765
UAT	63	4	45		112	9	13	31	27	32	610
Total	524	274	502	190	1490	98	167	278	416	531	7404

Table VI: Defect data of Project D

Defect Distribution by Origin For Project D						No. of Defects based on Criticality (i.e. Severity)					
<Phase>	Requirements Analysis	Design	Coding	Integration Testing	Total	Show Stopper	Critical	High	Medium	Low	Weighed Number of Defects
Analysis	405				405	25	36	72	125	146	1959
Design	35	225			260	11	27	52	59	111	1207
Coding with UT	33	39	133		205	14	19	35	59	78	988
Integration Testing	45	15	250	160	470	27	55	58	145	185	2211
UAT	51	6	35		92	7	11	27	22	25	507
Total	569	285	418	160	1432	84	148	244	410	545	6872

Table VII: Defect data of Project E

Defect Distribution by Origin For Project E						No. of Defects based on Criticality (i.e. Severity)					
<Phase>	Requirements Analysis	Design	Coding	Integration Testing	Total	Show Stopper	Critical	High	Medium	Low	Weighed Number of Defects
Analysis	427				427	27	42	75	130	153	2087
Design	45	252			297	6	31	58	70	132	1328
Coding with UT	27	31	120		178	11	12	29	54	72	823
Integration Testing	39	11	200	139	389	19	39	41	128	162	1753
UAT	27	4	18		49	2	7	14	11	15	259
Total	565	298	338	139	1340	65	131	217	393	534	6250

DEFECT METRIC ANALYSIS

Based on the defect data shown in the table 3 to table 7, following metrics were captured for each project. For each metrics (1) is used to calculate the efficiency.

- Defect Removal Efficiency (Requirement Analysis)
- Defect Removal Efficiency (Design)
- Defect Removal Efficiency (Testing)
- Overall Review (Inspection) Effectiveness
- Post Release Defect Density (PRDD)
- Defect Escape Rate

The size of the project is calculated in terms of effort in person days. For each of the parameter two baselines were set, namely, Organization Target and Project Target. The change or improvement is measured for each one of them during the progress made in development of Project A thru Project E.

The control charts were drawn by considering the defect data of each project. Organization Target parameter is set at the Upper Control Limit for each.

i) Defect Removal Efficiency (DRE) - Requirement Analysis

The Defect Removal Efficiency of Requirement Analysis phase for all the projects is shown in Table VIII.

Table VIII

Project	Actual Efforts in Person Days	Organization Target for DRE (Analysis) %	Project Target for DRE (Analysis) %	DRE (Analysis) %
Project A	3100	85	85	28.14
Project B	2960	85	85	47.80
Project C	3030	85	85	65.27
Project D	2980	85	85	71.18
Project E	3010	85	85	75.58

The Organization target for DRE of Requirement Analysis is 85% and the Project target is fixed at 85%. The control chart for the same is shown in Fig 1.

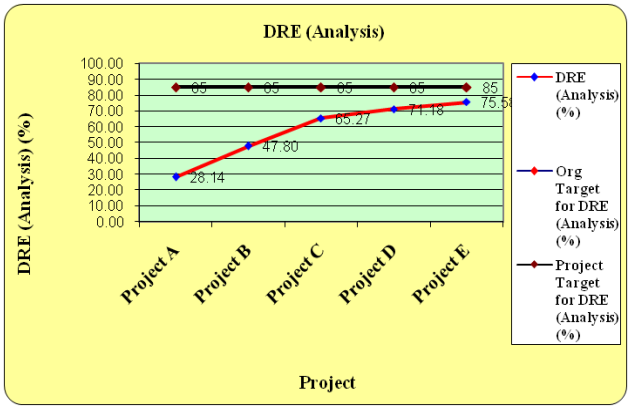

Figure 1

ii) Defect Removal Efficiency (DRE) – Design :

The Defect Removal Efficiency of Design phase for all the projects is shown in Table IX.

Table IX

Project	Actual Efforts in Person Days	Organization Taget for DRE (Design) %	Project Target for DRE (Design) %	DRE (Design) %
Project A	3100	85	85	16.14
Project B	2960	85	85	25.96
Project C	3030	85	85	52.63
Project D	2980	85	85	57.91
Project E	3010	85	85	68.12

The Organization target for DRE of Design is 85% and the Project target is fixed at 85%. The control chart for the same is shown in Fig 2.

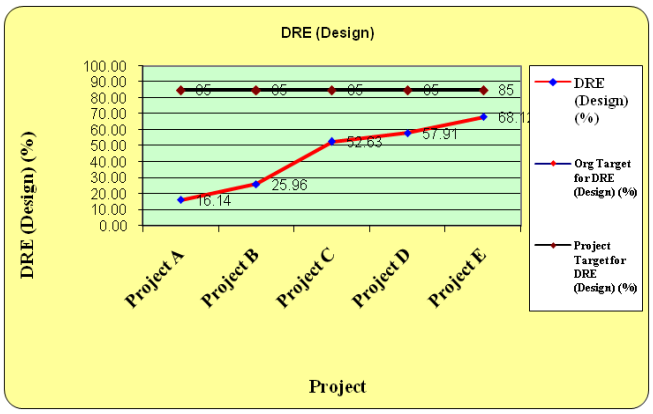

Figure 2

iii) Defect Removal Efficiency (DRE) – Testing :

The Defect Removal Efficiency of Testing phase for all the projects is shown in Table X.

Table X

Project	Actual Efforts in Person Days	Organization Taget for Testing Effectiveness %	Project Taget for Testing Effectiveness %	Testing Effectiveness %
Project A	3100	90	90	59.90
Project B	2960	90	90	72.64
Project C	3030	90	90	83.36
Project D	2980	90	90	83.63
Project E	3010	90	90	88.81

The Organization target for DRE of Testing is 90% and the Project target is fixed at 90%. The control chart for the same is shown in Fig 3.

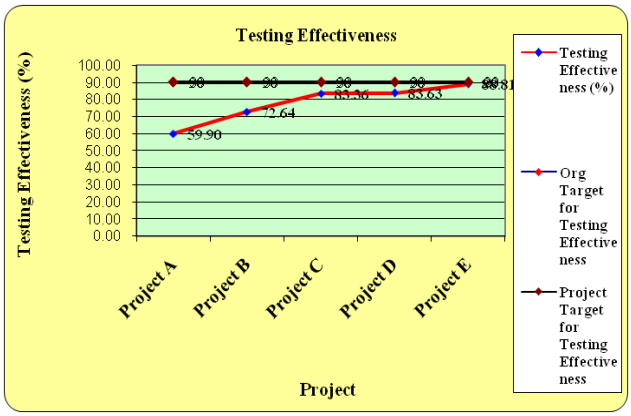

Figure 3

iv) Overall Review (Inspection) Efficiency :

The Overall Review or Inspection Efficiency for all the projects is shown in Table XI.

Table XI

Project	Actual Efforts in Person Days	Organization Taget for Review Effectiveness %	Project Taget for Review Effectiveness %	Review Effectiveness %
Project A	3100	80	80	32.19
Project B	2960	80	80	44.08
Project C	3030	80	80	54.83
Project D	2980	80	80	60.75
Project E	3010	80	80	67.31

The Organization target for Overall Review or Inspection is 80% and the Project target is fixed at 80%. The control chart for the same is shown in Fig 4.

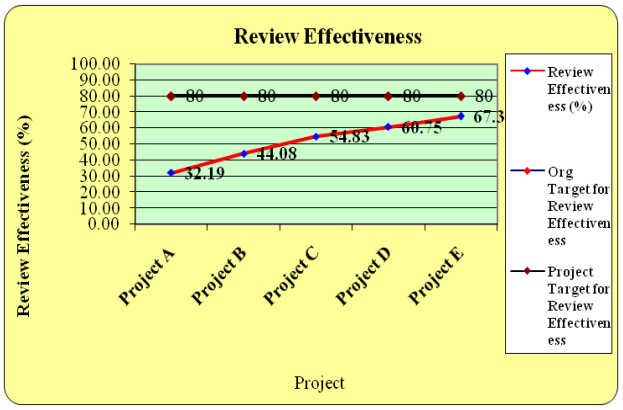

Figure 4

v) Post Release Defect Density (PRDD):

The Post Release Defect Density for all the projects is shown in Table XII.

Table XII

Project	Actual Efforts in Person Days	Organization Target for Post Release Defect Density	Project Target forPost Release Defect Density	Post Release Defect Density
Project A	3100	0.10	0.10	1.00
Project B	2960	0.10	0.10	0.44
Project C	3030	0.10	0.10	0.20
Project D	2980	0.10	0.10	0.17
Project E	3010	0.10	0.10	0.09

The Organization target for Post Release Defect Density is 10 (per person day) and the Project target is fixed at 10 (per person day). The control chart for the same is shown in Fig 5.

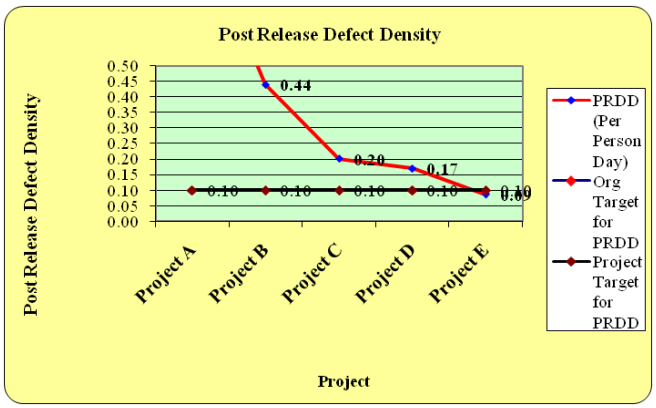

Figure 5

vi) Defect Escape Rate:

The Defect Escape Rate for all the projects is shown in Table XIII.

Table XIII

Project	Actual Efforts in Person Days	Organization Target for Defect Escape (%)	Project Target for Defect Escape (%)	Defect Escape (%)
Project A	3100	15	15	29.83
Project B	2960	15	15	14.78
Project C	3030	15	15	8.24
Project D	2980	15	15	7.38
Project E	3010	15	15	4.14

The Organization target for Defect Escape Rate is 15% and the Project target is fixed at 15%. The control chart for the same is shown in Fig 6.

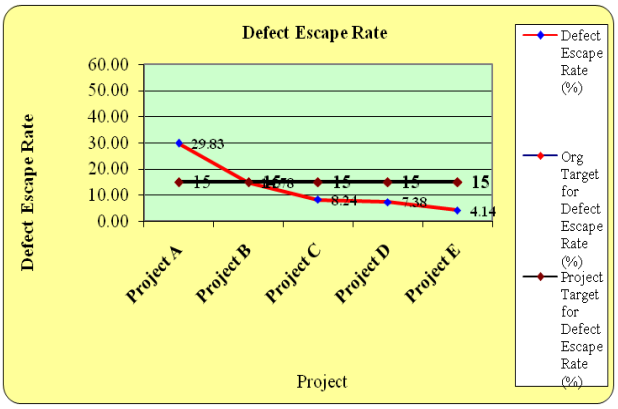

Figure 6

III. CONCLUSION

The review (both individual and team) process improved the efficiency of finding the defects that resulted in better efficiency even in testing activity that is tabulated in Table XIV.

Table XIV

	IE(RE)	IE(DE)	TE(UT)	TE(IT)
Project A	28.14	16.14	31.11	59.90
Project B	47.80	25.96	33.86	72.64
Project C	68.26	55.42	38.84	83.35
Project D	71.18	57.91	33.77	83.63
Project E	75.58	68.12	37.32	88.81

A progressive improvement in efficiency is recorded (based on the data presented in Section II) from Project A to Project E by improving the review process in each stage of development activity as depicted in Figure 7.

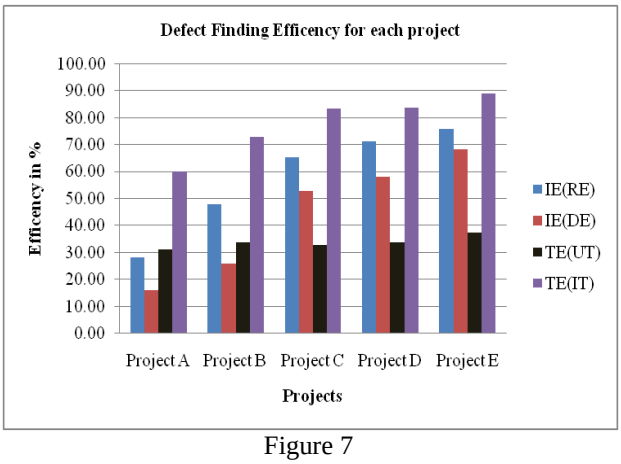

Figure 7

As a result of improved efficiency in defect detection, the count of the defects reduced from Project A to Project E which is tabulated in Table XV.

Table XV

	Project A	Project B	Project C	Project D	Project E
UAT Defects	565	261	112	92	49
Total Defects	2078	1706	1490	1432	1340

The User Acceptance Test (UAT) defects came down to 3.65% (for Project E), from 27% (for Project A). The quality in terms of fewer defects escaped from the development team to the UAT increased by adopting an improved method of inspection process as can be seen from Figure 8.

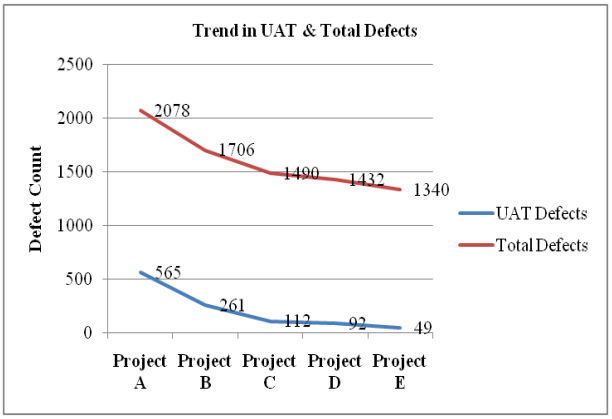

Figure 8

It is very important not to allow requirements anomalies, requirements errors, and requirements omissions to flow into the coding stage, because requirements problems cannot be found and removed by testing. Design problems should also be found prior to code development. The key point is that defects should be found within a very short span from the time of origination. Defects that originate in a specific phase such as requirements should never be allowed downstream into design and code.

In e-commerce projects, a critical aspect of product integration is the management of internal and external interfaces of the products and product components to ensure compatibility among the interfaces. These interfaces are not limited to user interfaces, but also apply to interfaces among components of the product, including internal and external data sources, middleware, and other components that may or may not be within the development organization's control but on which the product relies. Attention should be paid to interface management throughout the project.

REFERENCES

1. Kenneth C Laudon and Carol Guercio Traver, "E-Commerce – Business, Technology, Society",10th Edition, 2014, PEARSON, pp 3-11, pp 108-121.
2. E-Commerce and E-Business Belmont, en.wikibooks.org, December 29, 2013, pp. 5–9.
3. https://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_428597.pdf, para. 2, page 4 [Accessed: Aug 22, 2018]
4. Capers Jones, "Software Engineering Best Practices – Lessons from Successful Projects in Top Companies" 2010 Edition, McGraw-Hill, pp 124-128.
5. Joseph M. Juran, A. Blanton Godfrey, "Juran's Quality Handbook," Fifth Edition, McGraw-Hill . pp 2.5-2.12
6. https://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_428597.pdf, para. 2, page 4 [Accessed: Aug 22, 2018]

7. Stephen. H. Kan, "Metrics and Models in Software Quality Engineering", Second Edition, Pearson Education, 2004, pp 164-165
8. B. Clark and D. Zubrow, "How good is the software: a review of defect prediction techniques", Software Engineering Symposium, IEEE Computer Press, Pittsburgh, PA, 2001, pp. 1–35
9. R. Sudarshan, Dr. S. K. Srivatsa. (2018, September). "A study on Stages of Defects Injection and possible methods of their avoidance in E-commerce Web Sites". *IJAIEM* [Online]. pp. 14-24. Available: https://www.ijaiem.org/Volume7Issue9/IJAIEM-2018-09-23-5.pdf

AUTHORS PROFILE

Sudarshan R is a Bachelor of Engineering in Computer Science, and M. Tech in Information Technology. He has over 19 years of experience in IT industry that includes many CMMI companies. He has experience in the area of Software Delivery & Software Quality Assurance. He is pursuing his Phd in VELS University, Chennai.

Dr.S.K. SRIVATSA received the Bachelor of Electronics and Telecommunication Engineering degree from Jadavpur University, Calcutta, India. Master's degree in Electrical Communication Engineering and Ph.D from the Indian Institute of Science, Bangalore, India. He was a Professor of Electronics Engineering in Anna University, Chennai, India. His current research activities pertain to computer networks, Design and Analysis of algorithms, coding Theory and Artificial Intelligence & Robotics. He has produced seventy PhDs' and is the author of over 750 publications.