*International Journal of Application or Innovation in Engineering & Management (IJAIEM)*
**Web Site: www.ijaiem.org Email: editor@ijaiem.org**
**Volume 7, Issue 9, September 2018** **ISSN 2319 - 4847**

# A study on Stages of Defects Injection and possible methods of their avoidance in E-commerce Web Sites

**Sudarshan R[1] and Dr. S. K. Srivatsa[2]**

[1]Research Scholar, Department of Computer Science and Engineering,
VELS University, Chennai 600 117, India

[2]Former Senior Professor, Department of Computer Science and Engineering,
Anna University, Chennai 600 025, India

## ABSTRACT

***E-Commerce portal development includes a number of integrations like Vendor Information, Buyer Information and Authentication, Inventory, Payment Gateways, Delivery Logistics etc., hence it becomes very important to minimize defects at the end of each phase of developmental activity. Broadly the developmental activities include Requirement Development, Design, Coding and Testing. The methodology to develop the software that is considered is Iterative Rational Unified Process (RUP). This paper is an effort to look at the origin of defects in different stages of the software cycle and the possible avoidance methods in E-Commerce sites.***

**Keywords: :** E-Commerce, Defects, Cohesion, Dependency Matrix

## 1. INTRODUCTION

**E-Commerce** is the use of the Internet, the Web, and apps to transact business. More formally, digitally enabled commercial transactions between and among organizations and individuals. 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 India gradually the trend of doing electronic business is increasing and there are several software service companies claiming to develop and deliver E-Commerce web portals.

## 2. DEFECT PEDAGOGY

**Defect:** An imperfection or deficiency in a work product where that work product does not meet its requirements or specifications and needs to be either repaired or replaced, as per IEEE Std. 1044-2009.

According to Software Engineering Institute, a defect in software engineering parlance is any flaw or imperfection in a software work product or software process [2]. A software work product is any artifact created as part of the software process. Defect removal is one of the top expenses in any software project and it greatly affects schedules. Effective defect removal can lead to reductions in the development cycle time and good product quality.[3 The Text Book].

### 2.1 Defect removal process

To describe defect removal process clearly, we must first understand the activities in the development process that are related to defect injections. Defects are injected into the product or intermediate deliverables of the project at various stages. It is incorrect to assume that all defects are injected or introduced at the beginning of the software development life cycle. The defects appear only when specific operating conditions arise. Improved software reliability starts with understanding that the characteristics of software failures require analysis techniques distinct from those used for hardware reliability [5].

Table 1 shows an example of the activities in which defects can be injected and subsequent removal of the same for a

development life cycle. For the different phases before testing, various artifacts developed during the development cycle themselves are subject to defect injection as can be seen in Table 1. Reviews or Inspections at the end of each phase are crucial in defect removal as shown in Table1.

**Table 1:**Activities Associated with Defect Injection and Removal

| **Development Phase** | **Defect Injection Activity** | **Possible Defect Removal Activity** |
|---|---|---|
| Requirements Development | Requirements gathering process | Requirement analysis and review |
| Design (High-Level and Low-Level) | Design Work | High-Level and Low-Level Design Inspections |
| Code Implementation | Coding | Code Inspections |
| Integration/build | Integration and build process | Build verification testing |
| Unit Test | Bad fixes | Testing itself |
| Component Test | Bad fixes | Testing itself |
| System Test | Bad fixes | Testing itself |
| User Acceptance Test | Bad fixes | Testing itself |

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.

## 3. E-COMMERCE OVERVIEW

Various applications of e-commerce are continually affecting trends and prospects for business over the Internet, including e-banking, e-tailing and online publishing/online retailing. A more developed and mature e-banking environment plays an important role in e-commerce by encouraging a shift from traditional modes of payment (i.e., cash, checks or any form of paper-based legal tender) to electronic alternatives (such as e-payment systems), thereby closing the e-commerce loop [7].

### 3.1 A Few Advantages of e-Commerce

- Expanded Geographical Reach
- Expanded Customer Base
- Increase Visibility through Search Engine Marketing
- Provide Customers valuable information about the business
- Available 24/7/365 - Never Close
- Build Customer Loyalty
- Reduction of Marketing and Advertising Costs
- Collection of Customer's reusable, valuable data

### 3.2 Common E-Commerce modules

The **Rational Unified Process** (**RUP**) is an iterative software development process framework created by the Rational Software Corporation, a division of IBM [8]. The important modules of E-commerce system developed using RUP Model are [9] [10]:

1. Membership & Account Management
2. Catalog
3. Search
4. Browse & Shop

5. Pricing & Promotions
6. Order Management
7. Checkout & Payment
8. Shipping & Delivery
9. Integrations (both external and internal)

The development of each of the above modules is done with the Inception, Elaboration, Construction, and Transition process steps as defined by the RUP, which is iterative. Salient feature of each of the modules is listed below [5].

### 3.3 Membership & Account Management

- Membership Accounts can be created, managed and searched
- Create and view any returns
- View previous orders including status of orders and line items
- View account and invoice information including the ability for customers to pay for invoices by any of the online payment option or Cash On Delivery
- Manage, view and print recent transaction statements
- Request "email me" when back in stock
- Quick reorder from previous orders
- Create and save baskets

### 3.4 Catalog

- CSV and XML import and export of products, product data
- Image management, auto resizing and generation of thumbnails
- Manage master products, variations, related products and multiple price lists
- Integration with 3rd party warehousing system feeding live stock levels
- Define different product types such as gift vouchers, subscriptions, memberships, digital products and events

### 3.5 Search

- Fast site-wide searches returning accurate results from all content
- Multi-faceted search allows customers to sort, apply and remove filters on the results
- Sorting (price, name, new products, star rating, pre-defined metrics – or other fields specific to the context)
- Pagination (customer can also choose the number of products to display at any one time)
- Filtering (customer can filter the list of products by applying filter options)
- Promotional areas for advertisements and banners. These can be set to display against specific search terms and can be configured to display offers, products or even content pages

### 3.6 Browse & Shop

- Multi-faceted and layered navigation for advanced filtering and sorting based on user defined criteria for search results and for product categories
- Multiple product images with high resolution product image zooming capability
- Stock availability information is fetched and displayed from different vendor sites
- Product comparison information is facilitated by referring different vendor sites
- Product reviews and ratings are obtained from different sites and presented to the customer
- Relevant promotions and related product cross-sells information is fetched and displayed on product pages and in the shopping basket

### 3.7 Pricing & Promotions

- Handles complex pricing models that includes gross/net pricing, previous price, tax structure etc.
- Personalized shipping pricing
- Supports quantity based pricing
- Configurable rules based promotions and offers
- Session specific offers; can be triggered by a customer filing out a feedback form, referring a friend etc
- Discounts include, % off an order, % off a product(s)
- Buy x get y Free

- Free promotional products
- Free shipping
- Bundles: Buy 2 get 1 Free, 3 for Rs. 900 etc
- Product inclusions and/or exclusions based on the pattern of search
- Allowing promotional Coupons / vouchers
- Set start and end dates and timing for promotions

### 3.8 Order Management

- Powerful search to find orders quickly and easily based on multiple criteria
- Manage status and append notes to orders down to line item level
- User defined order workflows for handling different payment and shipping methods
- Print orders, invoices and packing slips
- Process returns and raise credit notes
- Report on abandoned baskets / orders
- Audit payment details such as paid or unpaid, payment method, receipt number, receipt value, transaction reference

### 3.9 Checkout & Payment

- Customizable, streamlined and guided checkout processes facilitating member and non-member checkouts
- Integrated with leading payment service providers
- Sophisticated 'My Account' feature with management of personal details, delivery addresses, saved baskets, active and past orders
- Shipping cost management

### 3.10Shipping & Delivery

- Weights / Units delivery calculations with upper and lower thresholds
- Integration with courier pricing matrixes and shipment tracking
- Exceptions for specific products which can have separate costs and different workflows for fulfillment
- Add delivery surcharges for individual expensive items that require additional cost to deliver

### 3.11Integrations

- Integrates with 3rd party applications such as ERP or CRM systems
- Integrates with multiple secured payment gateways
- Integrates with online chat software applications
- Integrates with Google sitemaps for location tracking
- Automatically updates vendor product stock levels

## 4. A LITERATURE STUDY ON DEFECT REMOVAL EFFECTIVENESS

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 [4]. A simple defect model is often an enumeration of development errors after they have occurred. But a software system that has had no security or reliability failures is not necessarily secure or reliable. The defects appear only when specific operating conditions arise [5].

### 4.1 Defect Removal Effectiveness

For the development phases before testing, the reviews or inspection at end-of-phase activities are the key vehicles for defect removal. For the testing phases, the testing itself is for defect removal, as depicted in Table1. When the defects found by testing are fixed incorrectly, there is another chance to inject defects. Defect removal effectiveness (DRE) for each development stage can be defined as [4]:

$$\frac{\text{Defects removed (at the stage)}}{\text{Defects existing} + \text{Defects injected}}$$

(On stage entry) (During development of the stage)

To derive an operational definition, Stephen H. Kan [4] 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.

**Table 2 :** Defect data Cross-tabulated by phase during which defect was found and its origin

| Defect Distribution by Origin | | | | | |
|---|---|---|---|---|---|
| <Phase> | Requirements Analysis | Design | Coding | Testing | Total |
| Requirement Analysis | 120 | | | | 120 |
| Design | 12 | 40 | | | 52 |
| Coding | 63 | 23 | 90 | | 176 |
| Component Testing | 10 | 12 | 90 | 45 | 157 |
| Integration Testing | 231 | 9 | 45 | | 285 |
| Total | 436 | 84 | 225 | 45 | 790 |

Let us look at the example in Table2. The matrix is triangular because the origin of a defect is always at or prior to the current phase. Coding effort includes Code Inspection and Unit Testing. Based on the definition given earlier (Section 4.1), we can calculate the various effectiveness metrics as below:

### 4.2 Requirements Inspection Effectiveness: IE(RE)

Defects removed at RE: 120
Defects existing on stage entry: 0
Defects injected in current phase (attributed to RE): 436

$$IE(RE) = \frac{120}{436} \times 100\% \quad (1)$$

$$= 27.53\%$$

### 4.2.1 Design Inspection Effectiveness: IE(DE)

Defects removed at DE: 52
Defects existing on stage entry: 436+84-120=400
Defects injected in current phase (attributed to DE): 84

$$IE(DE) = \frac{52}{436+84-120} \times 100\% \quad (2)$$

$$= 13\%$$

### 4.2.2 Code Inspection Effectiveness: IE(CD)

Defects removed at CD: 176
Defects existing on stage entry: 436+84+225-120-52=573
Defects injected in current phase (attributed to CD): 436

$$IE(CD) = \frac{176}{436+84+225-120-52} \times 100\% \quad (3)$$

$$= 30.72\%$$

### 4.2.3 Component Test Effectiveness: TE(CT)

Defects removed at CT: 157
Defects existing on stage entry: 436+84+225-120-52-176=397

Defects injected in current phase (attributed to CT): 45

$$TE(CT) = \frac{157}{436+84+225-120-52-176} \times 100\% \quad (4)$$

$$= 39.55\%$$

#### 4.2.4 Integration Test Effectiveness: TE(IT)

Defects removed at IT: 157
Defects existing on stage entry: 436+84+225-120-52-176=397
Defects injected in current phase (attributed to IT): 45

$$TE(IT) = \frac{285}{436+84+225+45-120-52-176} \times 100\% \quad (5)$$

$$= 64.48\%$$

## 5 AN ANALYSIS OF DEFECT PREVENTION IN E-COMMECRCE SITES

The e-commerce project development includes several modules and a large number of integrations as listed in 3.2. Each module is constructed under four phases of development life cycle of RUP. 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.

### 5.1 Project Case Study

This study has been performed on three e-commerce applications sequentially developed using ATG Framework. For the sake of reference identity they are called Project A, Project B and Project C. The size of each project was around 90 EKLOC. The average team experience in relevant technology is >5 year for all the three 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).

Each project's defect data (recorded for both Inspection and Testing) is presented below. A matrix approach (as in Section 4.1) 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 [11].

### 5.2 Data Analysis of *Project A*

**Table 3 :** Defect data of Project A, cross-tabulated by phase during which defect was found and its origin

| Defect Distribution by Origin For Project A | | | | | |
|---|---|---|---|---|---|
| <Phase> | Analysis | Design | Code with Unit Testing | Testing | Total |
| Analysis | 132 | | | | 132 |
| Design | 18 | 90 | | | 108 |
| Code with UT | 78 | 108 | 243 | | 429 |
| Testing | 10 | 12 | 363 | 459 | 844 |
| UAT | 231 | 122 | 212 | | 565 |
| Total | 469 | 332 | 818 | 459 | 2078 |

#### 5.2.1 Requirements Inspection Effectiveness: IE(RE)

Defects removed at RE: 120
Defects existing on stage entry: 0
Defects injected in current phase (attributed to RE): 436

$$IE(RE) = \frac{132}{469} \times 100\% \quad (6)$$

$$= 28\%$$

#### 5.2.2 Design Inspection Effectiveness: IE(DE)

Defects removed at DE: 108
Defects existing on stage entry: 469+332-132=669
Defects injected in current phase (attributed to DE): 332

$$IE(DE) = \frac{108}{469+332-132} \times 100\% \quad (7)$$

$$= 16.14\%$$

#### 5.2.3 Unit Test Effectiveness: TE(UT)

Defects removed at UT: 429
Defects removed at subsequent phases: 844+565

$$TE(UT) = \frac{429}{429+844+565} \times 100\% \quad (8)$$

$$= 23.34\%$$

#### 5.2.4 Testing (Integration) Effectiveness: TE(IT)

Defects removed at IT: 459
Defects removed at subsequent phases: 565

Defects injected in current phase (attributed to IT): 243

$$TE(IT) = \frac{844}{844+565} \times 100\% \quad (9)$$

$$= 60\%$$

### 5.3 Data Analysis of *Project B*

**Table 3** Defect data of Project B, cross-tabulated by phase during which defect was found and its origin

| Defect Distribution by Origin For Project B | | | | | |
|---|---|---|---|---|---|
| <Phase> | Analysis | Design | Code with Unit Testing | Testing | Total |
| Analysis | 261 | | | | 261 |
| Design | 21 | 108 | | | 129 |
| Code with UT | 48 | 63 | 251 | | 362 |
| Testing | 63 | 23 | 360 | 247 | 693 |
| UAT | 153 | 18 | 90 | | 261 |
| Total | 546 | 212 | 701 | 247 | 1706 |

#### 5.3.1 Requirements Inspection Effectiveness: IE(RE)

Defects removed at RE: 261
Defects existing on stage entry: 0
Defects injected in current phase (attributed to RE): 546

$$IE(RE) = \frac{261}{546} \times 100\% \quad (10)$$

$$= 47\%$$

#### 5.3.2 Design Inspection Effectiveness: IE(DE)

Defects removed at DE: 129
Defects existing on stage entry: 546+212-261=497
Defects injected in current phase (attributed to DE): 212

$$IE(DE) = \frac{129}{546+212-261} \times 100\% \quad (11)$$
$$= 25\%$$

#### 5.3.3 Unit Test Effectiveness: TE(UT)

Defects removed at UT: 362
Defects removed at subsequent phases: 693+261

$$TE(UT) = \frac{362}{362+693+261} \times 100\% \quad (12)$$
$$= 27\%$$

#### 5.3.4 Testing (Integration) Effectiveness: TE(IT)

Defects removed at IT: 693
Defects removed at subsequent phases: 261

Defects injected in current phase (attributed to IT): 247

$$TE(CT) = \frac{693}{693+261} \times 100\% \quad (13)$$
$$= 72.64\%$$

### 5.4 Data Analysis of *Project C*

**Table 4:** Defect data of Project C, cross-tabulated by phase during which defect was found and its origin

| Defect Distribution by Origin For Project C | | | | | |
|---|---|---|---|---|---|
| <Phase> | Analysis | Design | Code with Unit Testing | Testing | Total |
| Analysis | 342 | | | | 342 |
| Design | 33 | 207 | | | 240 |
| Code with UT | 36 | 45 | 154 | | 235 |
| Testing | 27 | 18 | 213 | 108 | 366 |
| UAT | 63 | 4 | 45 | | 112 |
| Total | 501 | 274 | 412 | 108 | 1295 |

#### 5.4.1 Requirements Inspection Effectiveness: IE(RE)

Defects removed at RE: 342
Defects existing on stage entry: 0
Defects injected in current phase (attributed to RE): 501

$$IE(RE) = \frac{342}{501} \times 100\% \quad (14)$$
$$= 68\%$$

#### 5.4.2 Design Inspection Effectiveness: IE(DE)

Defects removed at DE: 240
Defects existing on stage entry: 501+274-342=433

Defects injected in current phase (attributed to DE): 274

$$IE(DE) = \frac{240}{501+274-342} \times 100\% \quad (15)$$

$$= 55.42\%$$

### 5.4.3 Unit Test Effectiveness: TE(UT)

Defects removed at UT: 235
Defects removed at subsequent phases: 366+112

$$TE(UT) = \frac{235}{235+366+112} \times 100\% \quad (16)$$

$$= 32\%$$

### 5.4.4 Testing (Integration) Effectiveness: TE(IT)

Defects removed at IT: 366
Defects removed at subsequent phases: 112

Defects injected in current phase (attributed to IT): 108

$$TE(CT) = \frac{366}{366+112} \times 100\% \quad (17)$$

$$= 76.56\%$$

### 5.5 Case Study Inference

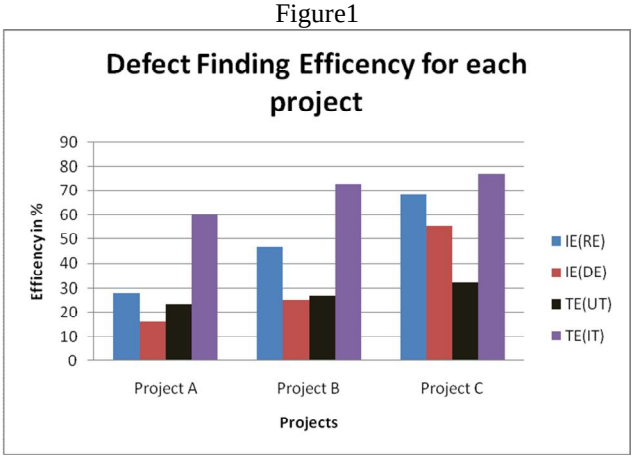
The review (both individual and team) process improved the efficiency of finding the defects that resulted in better efficiency even in Testing activity that is tabulated in Table 5.

**Table 5:** Defect finding efficiency of Inspection and Testing Activity

| | IE(RE) | IE(DE) | TE(UT) | TE(IT) |
|---|---|---|---|---|
| **Project A** | 28 | 16.14 | 23.34 | 60 |
| **Project B** | 47 | 25 | 27 | 72.64 |
| **Project C** | 68 | 55.42 | 32 | 76.56 |

A progressive improvement in efficiency is recorded from Project A to Project C by improving the review process in each stage of development activity as depicted in Figure1

Figure1




As a result of improved efficiency in defect detection the count of the defects reduced from Project A to Project C which

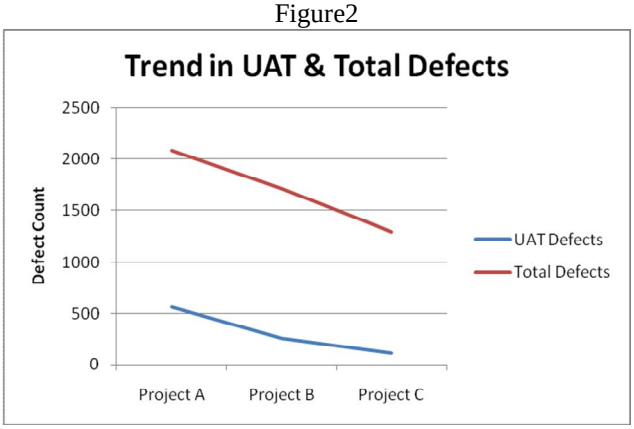
is tabulated in Table 6.

Table 6 : Count of defects in each of the project

| | Project A | Project B | Project C |
|---|---|---|---|
| UAT Defects | 565 | 261 | 112 |
| Total Defects | 2078 | 1706 | 1295 |

The User Acceptance Defects came down to 8% (for Project C), 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 2.

Figure2




The results suggest that products with very low levels of defects are associated with a disciplined approach to planning quality activities, especially static defect removal tasks. The omission of quality practices, such as inspections, can lead to defects that can exceed the capabilities of existing code analysis tools [12].

## 6.CONCLUSION

Defect prevention typically depends on analysis that extends beyond the practice of dynamic testing. The projects analyzed in Section 4.1, pointed to a disciplined lifecycle approach with quality defect identification and removal practices combined with code analysis to provide the strongest results for building a defect free software system.

Inspections have a synergistic relationship with other forms of defect removal such as testing and static analysis and also are quite successful as defect prevention methods. Recent work on software inspections by Tom Gilb, one of the more prominent authors dealing with inspections, and his colleagues continues to support the early finding that the human mind remains the tool of choice for finding and eliminating complex problems that originate in requirements, design, and other noncode deliverables [13]. Indeed, for finding the deeper problems in source code, formal code inspections still outrank testing in defect removal efficiency levels.

The various modules of an E-commerce website (as discussed in Section 3), have to be first analyzed for the "Dependency" on one another, to understand the *logical sequence* of artifacts (such as Design, Unit Test Cases) that have to be developed. For example, *Pricing and Promotion* module functionality depends on *Membership & Account Management* module, *Search* module depends on *Catalogue* and so on. Once all these dependencies are determined for all the modules, then the developmental activities like creation of Design Document, Unit Test Cases and the Static Analysis on them could be sequenced based on the logic flow. By this method several unknown scenario of flow that may cause defects could be unearthed and fixed or marked for future fixing [10].

**REFERENCES**

[1] Kenneth C Laudon, Carol Guercio Traver, "E-Commerce – Business, Technology, Society",10th Edition, 2014, PEARSON, pp 3-11.
[2] IEEE Standard Classification for Software Anomalies, IEEE Computer Society, IEEE Std. 1044 – 2009, 7 January 2010, pp 3-5.
[3] https://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_428597.pdf, para. 2, page 4 [Accessed: Aug 22, 2018]
[4] Metrics and Models in Software Quality Engineering, by Stephen H. Kan, Second Edition, Pearson Education,

2004, pp 159-160, pp 164-165, pp 167-169.
[5] Carol Woody, Robert Ellison, William Nichols,".Predicting Software Assurance Using Quality and Reliability Measures", in TECHNICAL NOTE, CMU/SEI-2014-TN-026, December 2014, pp 4-12.
[6] Metrics and Models in Software Quality Engineering, by Stephen H. Kan, Second Edition, Pearson Education, 2004, pp 164-165.
[7] http://en.wikibooks.org/wiki/Category%3AE-Commerce%20and%20E-Business , pp 23-25. [Accessed: Jun 06, 2018]
[8] https://www.ibm.com/developerworks/rational/.../1251_bestpractices_TP026B.pdf, pp 09-13 [Accessed: Aug 22, 2018]
[9] https://www.redtechnology.com/downloads/ecommerce-platform-feature-list.pdf. pp 1-9 [Accessed: Aug 22, 2018]
[10] http://www.ijettcs.org/Volume6Issue6/IJETTCS-2017-12-23-49.pdf. pp 163-167.
[11] B. Clark, D. Zubrow, "How good is the software: a review of defect prediction techniques", in: Software Engineering Symposium, IEEE Computer Press, Pittsburgh, PA, 2001, pp. 1–35.
[12] http://www.isixsigma.com/industries/software-it/defect-prevention-reducing-costs-and-enhancing-quality [Accessed: Sep 09, 2018]
[13] Capers Jones, "Software Engineering Best Practices – Lessons from Successful Projects in Top Companies" 2010 Edition, McGraw-Hill, pp 124-128.

**AUTHORS**

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 and Software Quality Assurance. Currently 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 Ph.D's and is the author of over 750 publications