IJCSE International Journal of Computer Sciences and Engineering Open Access

Review Paper Vol.-6, Issue-5, May 2018 E-ISSN: 2347-2693

Implications of Software Testing Strategies at Initial Level of CMMI: An Analysis

R. Sharma[1*], R. Dadhich[2]

[1] Department of CSI, University of Kota, Kota, India
[2] Department of CSI, University of Kota, Kota, India

[*]Corresponding Author*:* rekha_vivek@yahoo.com

Available online at: www.ijcseonline.org

Accepted: 18/May/2018, Published: 31/May/2018

Abstract - Software Testing is an essential and important phase of SDLC. The quality and acceptance of any software highly depends on the success of software testing phase. The successful completion of testing phase also ensures that the produced software is of good quality. In order to achieve high quality product lots of Process Maturity Models have been developed and CMMI is one of the most popular among them. The organizations at the initial level of CMMI (also non-CMMI compliance organizations) neither implement any of the standard processes for software product development nor they use any software testing strategies, hence, the quality of the product produced by them is always susceptible and imposes a great risk over its acceptance as well as on their survival. The main driving force behind this paper is to study the implications of software testing processes (partially or at introductory level) over the quality of software produced by the organizations at initial level.

Keywords- Software Testing Strategies, CMMI, Maturity Levels, Capability Levels, Process Areas

I. INTRODUCTION

In testing phase of Software Development Life-Cycle (SDLC), software is executed in order to verify that it is performing correctly in terms of logic as well as functionality; and most importantly that the software is built according to users' requirements and exhibits its intended purposes. In the present era of software development, the increasing complexity of infrastructure and supportive technology makes it almost impossible to develop a 100% defect-free product. The rapidly emerging technologies demand extra efforts as well as cost to develop software that supports diverse operating platforms running all over the world. Capability Maturity Model Integration (CMMI) is widely accepted as an Industry standard for the improvement of process used to develop and deliver software. It suggests various process areas grouped into two representations: Staged and Continuous. The implementation of standard CMMI is not feasible for small organizations and start-ups because it demands extra resources like time, budget and man-power. Since such organizations are already struggling to survive with limited resources, therefore, they strongly avoid implementing CMMI or any other process improvement models. The proposed study takes its motivation from the challenges faced by small organizations (more importantly by Start-ups) in developing and verifying the quality of software being developed in the absence of well-defined processes. This paper deals with the engineering process areas relevant to software testing only, which are available at maturity level 3 of the staged representation of CMMI and proposes a new model for Start-ups.

In this paper three key terms of software engineering are discussed i.e., a) software testing & strategies, b) CMMI & its representation, and c) CMMI Process Areas related to software testing. The complete paper is divided into 8 different sections. Section I introduces about the background of proposed work under study. Section II summarizes the related research work. Sections III, IV and V discuss Software Testing & Strategies, the CMMI & its representations, and Observed limitations of ML-1 respectively. The CMMI process areas related to software testing are discussed in section VI. The implications of software testing strategies at initial level of CMMI are presented in Section VII, whereas section VIII shows the conclusion of the analysis.

II. RELATED WORK

Many researchers have worked till now and are still working on Process Maturity Models. A. Luqman [4] conducted a study for the analysis & implementation of CMMI's Configuration Management (CM) process area in support category while adopting its continuous approach. His work suggested that the adoption of CMMI will drive the CM towards continuous process improvement of enterprise business for more reliable delivery of software products. P.

Monteiro et al. [5] worked to reunite validation and verification practices within CMMI maturity level 2. In his study, D. Singh [11] considered the factors such as software size and provided a benchmark for effort, quality, and cycle-time based on CMM Level 5 project data. S. R. Durugkar et al. [10] suggested that companies applying for CMMI level 2 assessments must find out the possibility of simultaneously implementing CMMI level 2 and the V & V Process Areas to produce high quality software.

R. Dadhich and U. Chauhan [6] analyzed the effect of integration of CMMI Level-3 in traditional software development process. In their work, they identified various risks associated with different phases of SDLC and proposed a risk mitigation plan by conducting a survey with 10 different development companies in India. R. K. Chauhan et al. [8] discussed various types of testing technique to measure attributes contributing for software quality. S. Jat et al. [14] conducted a study to assess a suitable technique for testing software and finding out errors. They realized that practically it is not possible to detect all the errors in software, but they suggested that Gray-Box testing technique is more accurate technique to find out errors and Black-Box testing requires minimum effort, cost and time to test a test-case.

III. SOFTWARE TESTING STRATEGIES

Software Testing is done to ensure that the developed product is error-free, produces the desired results, implements all requirements specifications into working functions, is easily maintainable, and meets customer expectations [3]. Software Testing has been defined in a number of ways in software Engineering literature. Some of the popular definitions are listed below:-

According to IEEE [1], Testing is a process which evaluates a system or its sub-system to verify that it satisfies its requirements as specified, either manually or by means of automated tools.

According to Myers [13], Software Testing is a process performed for the identification of errors in software.

In simplest form, testing can be defined as an activity performed for identification and correction of errors. The source of error plays a vital role in the successful and timely completion of a project. Different phases of SDLC may contain errors and if not handled properly, effect of error and cost of error-rectification increases drastically, which results in the failure of product. According to [13], the error that occurs during coding phase is known as a Bug. This error becomes a fault or defect, if the code produces wrong results either due to incorrectly implemented statement or due to absence of required statement. The error due to which a piece of code is unable to perform its intended purpose leads to a state known as product Failure. There are a number of symptoms demonstrating the state of failure, some of them are as follows:-

i) Code produces incorrect results,
ii) Under some circumstances, behaviour of code is not reliable.
iii) Code terminates abruptly,
iv) Code does not meet time and space requirements.

Hence, if not handled properly at correct place and time, an error or bug converts into a defect or fault which leads to the state of Failure. Testing phase of SDLC is responsible for handling all of the above mentioned terms.

Testing is a critical phase of software development activity and consumes approximately 40% - 50% of total project development efforts. Realizing the vital impact of testing on the quality of a product, it is necessary to develop and follow a software testing strategy to minimize the chances of failure. A software testing strategy [3] helps software developers in performing the software testing process in a planned and systematic manner. It incorporates test plans, the methods, techniques and tools, test case design, test specifications, test execution, resultant data collection and evaluation of collected data. In short, testing strategy provides the software developers with a road map which describes the steps to be conducted as a part of testing, and also determines the required effort, time, and resources. It must contain complete description of the test procedure along with the purpose and requirements of testing.

Depending upon the nature and size of the software being developed, a number of testing strategies have been proposed. All of these strategies provide a template [12] to the developer for testing and have the following characteristics in common:-

- Testing begins at the unit (component or module) level and proceeds outward to test the integrated units, and finally to test the entire system.
- Different testing techniques are used during different levels of testing, and also during different phases of software development.
- Either software developer or an Independent Test Group (in case of large projects) performs the testing process.
- Testing and Debugging are different activities and should not be used interchangeably, but debugging must be accommodated / incorporated in every testing strategy.

Low-level tests and high-level tests are the two basic ingredients for designing an effective testing strategy. Low-level tests are performed on small pieces of source codes and are required to ensure that the requirements are implemented correctly, whereas, high-level tests are necessary to validate major functions of software. It ensures that the software functions as per the user requirements. A strategy must provide sufficient guidance for the tester and a set of milestones for the manager. Software testing strategy produces a detailed document elucidating the entire test plan inclusive of all test cases used during the testing activity. This detailed document also enlists the weaknesses or discrepancies to be resolved prior to its future usages. A testing strategy is not only required for improving the software quality but also necessary for the improvement of procedures and plans used for testing purposes. It helps in documenting the issues and their rectifications [7] so that those issues can be avoided in upcoming tests.

IV. CAPABILITY MATURITY MODEL INTEGRATION (CMMI) & ITS REPRESENTATIONS

CMMI is a process level improvement and appraisal program (commonly referred as Software Process Improvement Maturity Model) developed at Carnegie Mellon University by a group of experts from Industry, government and Software Engineering Institute (SEI). CMMI is an improvement over its predecessor Capability Maturity Model, i.e., CMM for Software or Software CMM. CMMI is globally accepted as a software development standard. It supplies a set of guidelines for development process to improve the quality [1] of a software product. It is a compilation of best practices that could be implemented to improve the processes adopted to develop a software product so that the ultimate product becomes a high quality product and the organization meets it business objectives and goals feasibly. It [1] can also be used as a framework for appraising the process maturity of the organization.

The main aim for the development of CMMI was to integrate all the existing maturity models into a single framework [2] in order to improve their usability. The quality of process used to develop and maintain a product predominantly influences its quality, so prior to the emergence of CMMI; the objective of all maturity models popularly used at that time was on the improvement of processes used for the purpose of development. The special contribution of three models, namely, CMM for Software (SQ-CMM) v2.0 draft c [2], Electronic Industries Alliance Interim Standard (EIA/IS) / Systems Engineering Capability Model (SECM) [2], and Integrated Product Development CMM (IPD-CMM) v0.98 [2] is worth-noticing in the development of CMMI. These three models were selected for integration because of their successful adoption for improving processes in an organization. Initially, CMMI was developed solely for Software Engineering, but now it provides process improvement and development guidelines for development of hardware products, for the delivery of all kind of services, and for the Acquisition of Product and Services. Currently three CMMI models [1] are available–

i) CMMI for Acquisition
ii) CMMI for Development
iii) CMM for Services

"CMMI for Development" covers activities appropriate for the development of products and services. It contains practices that cover software engineering, systems engineering, hardware engineering, process management, product management, and other supporting processes used in development and maintenance.

According to SEI [1] CMMI -

(i) Integrates traditionally separate organizational functions into a single function
(ii) Sets process improvement goals and priorities for an organization,
(iii) Provides guidance for developing quality processes, and
(iv) Provides a reference point for process appraisal.

The representation of CMMI model allows an organization to pursue different improvement objectives. CMMI [2] can be represented in two ways:-

A. Staged Representation
B. Continuous Representation

A. Staged Representation: It is used widely as a standard representation to improve overall maturity of an organization. It provides a ladder-styled process improvement sequence. It specifies a systematic improvement path for process maturity across an organization by using a set of predefined process areas [2]. It uses 5 maturity levels to portray the overall status of the process exercised within the organization. These 5 maturity levels are organized as a set of layers where each lower layer provides a basis for the immediate next upper step. Each maturity level is comprised of specific and generic practices for its relevant process areas, and guides an organization in maturing the sub-processes associated with that particular maturity level so that the organization can move upwards to the next stage of maturity. The maturity levels (MLs) are numbered from 1 to 5 [2] as explained below:-

1. **ML-1: Initial:-** Organizations in this level lack the stable environment required to support processes. Although they use processes to develop and maintain products, but these processes are designed on ad hoc basis and are not maintained for reuse. Products developed by such organizations do succeed in terms of customer satisfaction but they are usually over-budgeted and deviate from their original time schedules and plans.

2. **ML-2: Managed:-** Organizations at this level are more organized, they have pre-planned process and work-schedule to develop & maintain products. The overall plan contains the details of work products, milestones and services. At any point of time, the overall status of project is visible to the management. In case of any deviation or delay, the management can take necessary actions to control the progress of project. Besides, the used processes are maintained for future reuse.
3. **ML-3: Defined:-** The above two levels follow reactive approach whereas this level uses proactive approach to manage processes. All the processes maintained at maturity level 2 are re-defined in more details at maturity level 3 in order to convert them into a set of standard processes. The organization derives the processes for the development, maintenance and delivery of futuristic product from its standard processes. The processes are defined for each procedure, tool, method & standard and are consistent across the organization. A minor trade-off between standard and project-specific process may exist because of the nature and requisites of project.
4. **ML-4: Quantitatively Managed:-** Organizations willing to achieve this level select the sub-processes that remarkably improve the overall performance of process. These selected sub-processes are subjected to some statistical techniques to obtain quantitative results suitable for establishing and managing process performance and quality benchmarks. These statistically managed processes make it possible to predict the performance of a process quantitatively rather than qualitatively as is the case in ML3.
5. **ML-5: Optimizing:-** Organizations interested to achieve this level continually improve their processes performance quantitatively by identifying the factors responsible for variation in processes. The improvements are facilitated by implementing technological innovations and by understanding the inter-dependencies and inter-relationships exist among various sub-processes of a process so that the sources of variation can be mitigated effectively.
6.

B. **Continuous Representation**: The organizations willing to improve the capability of a specific process use this representation of CMMI. This representation allows an organization to select the desired process area for improvement and suggests the sequence of improvements that best suits to achieve the business objectives of organization. It has the following 6 (six) capability levels to characterize improvement relative to an individual process area :-

1. **Level 0 – Incomplete**: The organization implements only some applicable specific practices.
2. **Level 1 – Performed**: The organization lacks the necessary processes for sustaining service levels.
3. **Level 2 – Managed**: The organization manages and reacts, but is not able to strategically predict costs of services and compete with lean competitors.

4. **Level 3 – Defined**: The organization anticipates changes in its environment and plans, but still lacks the ability to forecast changing costs and schedules of services.
5. **Level 4 – Quantitatively Managed:** The organization statistically forecasts and manages performance against selected cost, schedule, and customer satisfaction levels.
6. **Level 5 – Optimizing**: The organization can reduce operating costs by improving current process performance or by introducing innovative services to maintain their competitive edge.

As CMMI provides well-defined and proven Process Areas (PAs) for each and every phase of SDLC, Software Testing is not an exception. In CMMI terminology the terms Verification and Validation has replaced the term Software Testing. As we know that Testing is a process by which we can ensure the correctness, completeness and quality of a product, the processes related to verification and validation can be considered as the processes for software testing also. For better understanding, let us consider an emerging organization which is at the very initial level of development (say, a Software Start-up Venture) and does not have standardized processes at their disposal to work in a systematic and controlled manner in order to produce and deliver a product that satisfies user requirements and is built within time and budget limits. If such organization manage to invest some additional efforts and money to improve their products' quality (and hence its success rate also) and develop or outsources a proven software testing strategy, implements it, and regulates its working according to the outcomes of testing process, they will definitely have an edge over their counterparts not employing any plans and procedures for software testing. Once implementing a software testing strategy at the infantry age of their organization, they will succeed rapidly in the market by providing low-budgeted (or slightly over-budgeted), good quality product that satisfies customer requirements.

V. OBSERVED LIMITATIONS OF ML-1

According to the current definition of ML-1of CMMI, the software is developed in an informal, unplanned, uncontrolled, intuitive and non-systematic manner, hence, there is no defined testing process. Due to this various consequences can be identified, some of them are as follows:-

(i) The process adopted for developing product is unpredictable and changes frequently depending on the competence of individual performing it.

(ii) There are no well-defined processes or steps to develop and hence, to monitor the development activity. The developers are unable to tell the actual amount or percentage of work completed or the milestone achieved.
(iii) They are working as per their intuition and experience which creates a chaotic scene as a result, the time schedule, budget and delivery dates cannot be specified in exact.
(iv) This type of development process may or may not produce the product of desired quality and within time & budget constraints.
(v) The delivered product may or may not satisfy customer requirements as there is no quality check before delivering the product. This approach makes the software development a risky process.

VI. SOFTWARE TESTING PROCESS AREAS

CMMI directly does not provide process areas for software testing as the term software testing is not used in CMMI. The Process Areas related to software testing are provided at Maturity Level -3 (ML-3) of staged representation of CMMI. Among the process areas at ML-3 [2], the three Engineering Process Areas, namely, Process Integration (PI), Validation (VAL), and Verification (VER) are primarily concerned with software testing. CMMI process areas are defined by a set of goals and practices, grouped in two categories as under-

1) Generic Goals (GG) and Practices (GP) – They are common for all process areas.
2) Specific Goals (SG) and Practices (SP) – They are applicable for a specific process area only.

Both the generic and specific goals and practices applicable to specific process area must be implemented completely in order to qualify the concerned Process Area. This paper primarily deals with the specific practices and goals [2] for two of the above mentioned three process areas:-

A. **Validation (VAL):** The validation process provides as answer to the question – “Are we building the right product?” The purpose of Validation is to show that the product (or component) code functions as per expectations and fulfils its intended purpose when executed in its target environment. The actual products entitled for validation include a unit of code, an individual module, an integrated module, a complete system or product.

Specific Practices by Goals:

- **SG 1 Prepare for Validation**
  - **SP 1.1** Select Product for Validation
  - **SP 1.2** Establish the Validation Environment
  - **SP 1.3** Establish Validation Procedures and Criteria

- **SG 2 Validate Product or Product Component**
  - **SP 2.1** Perform Validation
    - **SP 2.2** Analyze Validation Results

B. **Verification (VER)**: The Verification is performed to obtain an answer to the question – “Are we building the right product?” In other words, this process checks whether the selected work-product conforms to its specifications or not. The work products entitled for verification include all the documents designed prior to coding phase such as Requirements Specification, Architectural Design. Detailed Design, Database Design etc.

Specific Practices by Goals:

- **SG 1 Prepare for Verification**
  - **SP 1.1** Select Work Product for Verification
  - **SP 1.2** Establish the Verification Environment
  - **SP 1.3** Establish Verification Procedures and Criteria

- **SG 2 Perform Peer Reviews**
  - **SP 2.1** Prepare for Peer Reviews
  - **SP 2.2** Conduct Peer Reviews
  - **SP 2.3** Analyze Peer Review Data

- **SG 3 Verify Selected Work Products**
  - **SP 3.1** Perform Verification
  - **SP 3.2** Analyze Verification Results

In short, Validation is a dynamic process of testing where product is placed under operating environment to decide whether it is working as intended or not, i.e., the product is developed correctly or not. Also, whether the developed product does what it supposed to do. Various methods of testing like White-Box, Black-Box etc. are used for this purpose. Verification is done prior to Validation. It is a human based method for static testing of documents, designs and codes (without executing) through which the selected work product is reviewed to ensure that it meets its requirements as specified and the correct product is being developed. Methods like inspections, walk-through etc. are used for the verification purposes. The successful completion of Validation and Verification ensures that the developed product is a quality product and will not fail during testing at user’s end.

VII. IMPLICATIONS OF SOFTWARE TESTING AT INITIAL LEVEL OF CMMI

A. Driving forces behind the Study

In present scenario, every year all over the world, thousands of Start-up ventures come into existence but very shortly most of them cease to exist because of their informal, unmanaged and unprofessional manner of working. Theses start-ups experience financial problems due to which they cannot afford for the formalism of processes at their initial stages. They cannot implement maturity models at their beginnings because it incorporates a lot of cost and efforts which is not available and affordable at that situation. They start their business with very small budget and very less man-power.

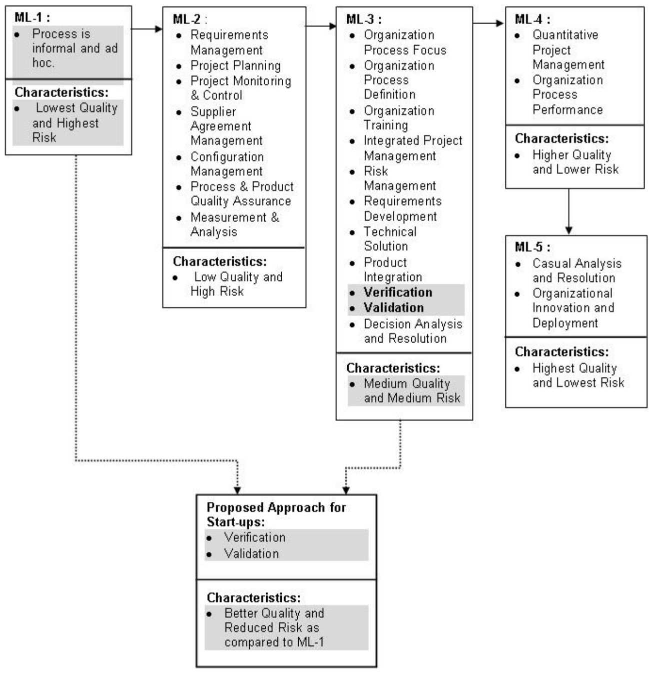
B. Proposed Study

Despite of the constraints of finance and efforts, if such organizations take some extra measures to plan not each and every process but at least the most critical, success-deciding, and quality reflecting process, i.e., the software testing process of SDLC at ML-1 of CMMI, then, they can produce and deliver products that are of high quality, low in cost (although slightly more than the product cost build with unplanned tests), built within time-bounds and satisfy customer requirements and expectations as well.

With the help of partial integration of VAL and VER process areas in ML-1, such organizations can better survive in the development market and can grow as an organization which can develop good quality products without implementing any maturity model in its entirety. This blend of ML-1 with a part of ML-3 will provide them with an edge over their competitors who are already performing well in the market, in terms of low-budgeted, high quality and timely delivered product as shown in Fig.1.

Some Certification organizations may argue that they can implement the Continuous representation of CMMI and achieve Capability levels for testing process only rather than implementing the Staged representation model. In that case also, the organization (in question) has to implement it level-wise without skipping any intermediate levels (same as the case with staged model) which is again a time and effort consuming task.

One of the important outcome of this study is for Software developers who are working on low-cost projects i.e., start-ups. They can gain benefit by following such approach which places them a bit above the ML-1 and ML-2 organizations, but not equivalent to an ML-3 organization, because it implements a small part of ML-3 process areas skipping the time, cost and effort consuming procedure required to implement the ML-2 Process Areas in its entirety.



ML-1 :
Process is informal and ad hoc.
Characteristics:
Lowest Quality and Highest Risk
ML-2 :
Requirements Management
Project Planning
Project Monitoring & Control
Supplier Agreement Management
Configuration Management
Process & Product Quality Assurance
Measurement & Analysis
Characteristics:
Low Quality and High Risk
ML-3 :
Organization Process Focus
Organization Process Definition
Organization Training
Integrated Project Management
Risk Management
Requirements Development
Technical Solution
Product Integration
Verification
Validation
Decision Analysis and Resolution
Characteristics:
Medium Quality and Medium Risk
ML-4 :
Quantitative Project Management
Organization Process Performance
Characteristics:
Higher Quality and Lower Risk
ML-5 :
Casual Analysis and Resolution
Organizational Innovation and Deployment
Characteristics:
Highest Quality and Lowest Risk
Proposed Approach for Start-ups:
Verification
Validation
Characteristics:
Better Quality and Reduced Risk as compared to ML-1

Figure 1. Proposed Structure of ML-1.

VIII. CONCLUSION

Not all organizations adopt CMMI [9] either due to the high implementation cost of CMMI or because of the small size of the organization. But they do use some other Software Process Improvement models.

This study (mainly focuses on Start-ups) suggests that if anyhow, the start-ups afford to formalize only the Process Areas related to Validation and Verification (even partially) then also they can increase their survival and success rate remarkably. Also, in future, if they want to implement CMMI maturity levels in their organization, they can do it easily without repeating the entire process for the above-mentioned process areas of ML-3.

REFERENCES

[1] "*Capability Maturity Model® Integration (CMMI®) Overview*", Carnegie Mellon University/ Software Engineering Institute, pp: 393-411, 2005.

[2] "*CMMI® for Development, Version 1.3*", Technical Report, CMMI Product Team, Carnegie Mellon University/ Software Engineering Institute, Nov-2010.

[3] R. S. Pressman, "*Software Engineering: A Practitioner's Approach*", Fourth Edition, McGraw-Hill Publications, pp: 448-512, 1997.

[4] A. Luqman, "*Implementation And Analysis of CMMI's Configuration Management Process Area; Applicable to "Defined" Level – 3*", International Conference of Information and Communication Technology, Karachi, Pakistan, IEEE Conference Publications, pp: 296-301, 27-28 Aug'2005.

[5] P. Monteiro; R. J. Machado, R. Kazman, "*Inception of Software Validation and Verification Practices within CMMI Level 2*", ICSEA '09. Fourth International Conference on Software Engineering Advances, Porto, Portugal, IEEE Conference Publications, pp: 536-541, Conference of Proceedings published by IEEE Computer Society, 20-25 Sep'2009.

[6] R. Dadhich, U. Chauhan, "*Integrating CMMI Maturity Level-3 in Traditional Software Development Process*", International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.1, pp: 17-26, January 2012.

[7] S. H. Trivedi, "*Software Testing Techniques*", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 10, pp: 433-439, October 2012.

[8] R. K. Chauhan, I. Singh, "*Latest Research and Development on Software Testing Techniques and Tools*", INPRESSCO, International Journal of Current Engineering and Technology, Vol.4, No.4, pp: 2386-2372, Aug 2014.

[9] M. Staples, M. Niazi, R. Jeffery, A. Abrahams, P. Byatt, R. Murphy, "*An exploratory study of why organizations do not adopt CMMI*", Journal of Systems and Software, Vol. 80, No.6, pp: 883-895, June'2007.

[10] S. R. Durugkar, V. Surwase, "*Software Validation and Verification Practices in CMMI levels*", BIOINFO Soft Computing, Vol. 1, Issue 1, pp: 01-04, 2011.

[11] D. Singh, "*Software Testing using CMMI Level 5*", International Journal of Computer Science Trends and Technology, pp: 233-242, Mar-Apr 2016.

[12] N. Chauhan, "*Software Testing: Principles and Practices*", Second Edition, Oxford University Press, India, Dec'2016, ISBN-13: 978-0198061847.

[13] R. Chopra, "*Software Testing (A Practical Approach)*", Third Edition, S.K. Kataria & Sons, India, 2010, ISBN: 9788189757908, 8189757903.

[14] S. Jat, P. Sharma, "Analysis of Different Software Testing Techniques", International Journal of Scientific Research in Computer Science and Engineering (IJSRCSE), Vol.5, Issue.2, pp.77-80, April '2017.

Authors Profile

R. Sharma pursued Bachelor of Science in 1988 and Master of Computer Applications in 1991 from Banasthali Vidyapeeth, Rajasthan, India She is currently pursuing Ph.D. and currently working as Guest Faculty in Department of Computer Science & Informatics, University of Kota, Kota, Rajasthan, India since 2013. Her main research work focuses on SoftwareTesting and CMMI. She has 24 years of teaching experience.

R. Dadhich is working as Professor in the Department of Computer Science and Informatics, University of Kota, Kota. She has more than 19 years of teaching and 11 years of research experience. Her areas of research are " Wireless Ad-Hoc Networks, Wireless Sensor Networks and Software Testing". She has visited University of Moratuwa, Colombo, Sri Lanka for research assignment. She is working as an Advisory/Editorial Board Member/Reviewer/Technical Committee member of various International/National Journals and Conferences. She has more than 60 research publications in Journals of international and national repute and 09 candidates completed their Ph.D. work with her. She also has authored 03 book-chapters and Edited 05 books. She is honoured as Expert member of Research Committees of different Universities. She has delivered expert lectures/invited talks and chaired technical sessions in many conferences/workshops & short term courses.