International Journal of Research, Science, Technology & Management
ISSN Online: 2455-2240, Volume 13 Issue III, December 2018

Organizational practices that effects Software Quality In Software Engineering process

Rohit Saxena

ABSTRACT-

Software quality metrics are a subset of software metrics that focus on the quality aspects of the product, process, and project. In general, software quality metrics are more closely associated with process and product metrics than with project metrics. Nonetheless, the project parameters such as the number of developers and their skill levels, the schedule, the size, and the organization structure certainly affect the quality of the product. Software quality metrics can be divided further into end-product quality metrics and in-process quality metrics. The essence of software quality engineering is to investigate the relationships among in-process metrics, project characteristics, and end-product quality, and, based on the findings, to engineer improvements in both process and product quality. Moreover, we should view quality from the entire software life-cycle perspective and, in this regard, we should include metrics that measure the quality level of the maintenance process as another category of software quality metrics. In general the organization process related to software development is always a complex task to be fit into any of the quality metrics. Analytically a complete software development organizational process can be achieved as a combination of one or more software quality metrics. But some of the organizational processes which do not come under software quality metrics lead to degrade the software quality.

Organization practices are performed as regular operations at various levels in the organization. These practices should be addressed continuously to run the process smoothly as predetermined. A Software development organization maintains metrics, time - work controls, Finance control tools and some other techniques to improve and finely tune the processes. But some of the software organization processes which cannot come under the vision of such tools make the development process unknowingly inefficient. Some of the Organization processes which cannot come under these control tools and metrics are effecting the Software development and maintenance life cycle at various levels.

I. INTRODUCTION

Software engineering is a complex engineering activity. It involves interactions between people, processes, and tools to develop a complete product. In practice, commercial software development is performed by teams consisting of a number of individuals ranging from the tens to the thousands. Often these people work via an organizational structure reporting to a manager or set of managers. The intersection of people, processes and organization and the area of identifying problem prone components early in the development process using software metrics have been studied extensively in recent years. Early indicators of software quality are beneficial for software engineers and managers in determining the reliability of the system, estimating and prioritizing work items, focusing on areas that require more testing, inspections and in general identifying problem spots to manage for unanticipated situations. Often such estimates are obtained from measures like code churn, code complexity, code coverage, code dependencies, etc. But these studies often ignore one of the most influential factors in software development, specifically people and organizational structure.

This interesting fact serves as our main motivation to understand the intersection between organizational structure and software quality Organizations that design systems are constrained to produce systems which are copies of the communication structures of these organizations. Similarly, it is also very important that the product quality is strongly affected by organization structure. With the advent of global software development where teams are distributed across the world the impact of organization structure for development process and its implications on quality is significant. To the best of our knowledge there has been little or no empirical

evidence regarding the relationship or association between organizational structure and direct measures of software quality like failures. In this paper we investigate these reasons between organizational practices and software quality by analyzing various facts at software engineering process.

II. LITERATURE SURVEY

From Basic Principles and Concepts for Achieving Quality the Software Quality Framework (SQF) developed in the early 1980s for the Department of Defense (DoD) by Baker and colleagues describes the conceptual elements necessary for building quality into systems, or any entity, and evaluating the quality actually achieved. They explained definitions and conceptual elements within the context of Capability Maturity Model Integration (CMMI) to show how CMMI codifies the concepts. Statistical process control (SPC) is the application of statistical methods to the monitoring and control of a process to ensure that it operates at its full potential to produce conforming product. Under SPC, a process behaves predictably to produce as much conforming product as possible with the least possible waste. It examines a process and the sources of variation in that process using tools that give weight to objective analysis over subjective opinions and that allow the strength of each source to be determined numerically. Variations in the process that may affect the quality of the end product or service can be detected and corrected, thus reducing waste as well as the likelihood that problems will be passed on to the customer.

According to various studies (see McConnell 1993 and references therein) the single most effective means to identify and correct code defects is the review process and focuses on design and code reviews, although formal reviews can be applied to any stage of software development. The most formal variety of reviews-inspections-typically catch about 60% of the defects present in software (Jones 1986). While these rates are generally much better than that achieved by simply testing the end product, the kinds of errors found with each method are often quite different. A significant difference between inspections and testing, though, is that defects are identified and corrected in one step. Testing simply identifies a defect-it reveals nothing about how to fix it. Formal design and code inspections provide an effective means to catch defects early, where they are easiest and cheapest to fix.

The organization's process asset library is a collection of items maintained by the organization for use by the people and projects of the organization. This collection of items includes descriptions of processes and process elements, descriptions of lifecycle models, process tailoring guidelines, process-related documentation, and data. The organization's process asset library supports organizational learning and process improvement by allowing the sharing of best practices and lessons learned across the organization. The organization's set of standard processes is tailored by projects to create their defined processes. The other organizational process assets are used to support tailoring as well as the implementation of the defined processes. The work environment standards are used to guide creation of project work environments.

A standard process is composed of other processes (i.e., sub processes) or process elements. A process element is the fundamental (e.g., atomic) unit of process definition and describes the activities and tasks to consistently perform work. Process architecture provides rules for connecting the process elements of a standard process. The organization's set of standard processes may include multiple process architectures. The organizational process assets may be organized in many ways, depending on the implementation of the Organizational Process Definition process area. Descriptions of lifecycle models may be documented as part of the organization's set of standard processes, or they may be documented separately. The organization's set of standard processes may be stored in the organization's process asset library, or they may be stored separately. A single repository may contain both the measurements and the process-related documentation, or they may be stored separately.

The term "software process improvement" denotes the "changes implemented to a software process that bring about improvements" [Olson et al. 1989]. ISO 9000-1 defines a process as "a set of interrelated resources and activities which transform inputs into outputs. ... Resources may include personnel, finance, facilities, equipment, techniques and methods" [ISO 9000-1 1994]. Correspondingly, a software process can be defined as "a set of activities, methods, practices, and transformations that people use to develop and maintain software and the associated products (e.g. ,project plans, design documents, code, test cases, and user manuals)" [Paulk et al. 1993a]. The intent of software process improvement is improving software product quality, increasing productivity, and reducing the cycle time for product development [Paulk et al. 1993b]. Thousands of software companies worldwide have established initiatives to improve software development performance. In a study of 119 German software companies that have implemented an ISO 9000 based quality system 88 % of the respondents say that they consider the improvement initiative successful[Stelzer, Taube 1998]. Several authors [Curtis 1997; Mattingly, Abreo 1997; Morrell, van Asseldonk 1997; Myers 1996;Quinn 1998; Siddall 1997; Stelzer et al. 1997] suggest that the problems software companies encounter while implementing process improvement initiatives might result from insufficient organizational change management. Therefore, various authors have stressed the importance of organizational change in software process improvement programs [Bottcher 1997; Curtis 1997; Humphrey 1989;

Humphrey 1997; Jackson 1997; Mattingly, Abreo 1997; Morrell, van Asseldonk 1997; Myers 1996; Myers 1997; Quinn 1998; Siddall 1997; Stelzer et al. 1997]. Some authors[Coffman, Thompson 1997; Goldenson, Herbsleb 1995; Stelzer et al. 1997] indicate that software organizations usually underestimate the efforts needed to accomplish the change process. McGuire states that "..even established and successful software development teams may experience uncertainty and disruption when beginning a software process improvement effort ..." [McGuire 1996a].

Pankaj Jalote in one of his publications mentioned that a high maturity organization is expected to use metrics heavily for process and project management. A study was conducted to understand how some of the high maturity organizations use metrics and summarizes the similarities found in the use of metrics, focusing on the metrics infrastructure employed, use of metrics in project planning, use of metrics in monitoring and controlling a project, and use of metrics for the improvement of the overall process.

There are many standards available to help organizations starting a measurement program. In our case, the ISO/IEC 15939 standard for establishing software measurement programs served as our main guideline. This standard, based on the Practical Software Measurement (PSM) methodology developed by McGarry et al. describes a four-phase process for implementing a measurement program. The process starts by establishing Management commitment and assigning roles and responsibilities to organizational units and individuals. Then, the metrics program is planned by deriving the set of metrics based on business goal. Next, in step three the program is implemented and the first results are obtained. Finally, in step 4 the program is evaluated by validating the implemented metrics and making changes accordingly. In this paper we have presented various organizations processes in software engineering that are over looked as a major parameter in capturing as metrics.

III. ORGANIZATION PRACTICES THAT EFFECTS SOFTWARE QUALITY

Always we feel few things are minor but they have impact on the environment. The effect of these facts become major when they are implemented repetitively, which introduces gaps between various artifacts that constructs the whole environment and process cycle.

Organization practices are performed as regular operations at various levels in the organization. These practices should be addressed continuously to run the process smoothly as predetermined. A Software development organization maintains metrics, time - work controls, Finance control tools and some other techniques to improve and finely tune the processes. But some of the software organization processes which cannot come under the vision of such tools make the development process unknowingly inefficient. Some of the Organization processes which cannot come under these control tools and metrics are effecting the Software development and maintenance life cycle at various levels.

They are:

1. Allocation of Human Resources
2. Maintenance of Documents and other Resources
3. Dependency with Core-Product team
4. Continuously changing the Project Managers
5. Miscellaneous factors

Allocation of Human Resources:

Resources that are shifting from one project to another

A common practices in resource allocation in software company is that a Resource management group will allocate and de- allocate resources based on the there designation and Qualification. A resource with good rating of performance will always paced in new project and keep on moving that person who will be involved key modules of each project. The resources will be changed also basing on the local product experience that is gained by him and will always be kept in key position as he goes on gaining more experience on the product or integrated development environment and other dependent environments. In the due course, Knowledge is transferred to the new resources that are invited to that project.

Improper Knowledge Transfer given the newly allocated resources.

When new resource person enters into the project irrespective of his knowledge the project information will be transferred based on existing person's experience about the work. The new comer receives the information given as it is, instead of studying its pros and cons. So the new comer will continue to work on the same and transfer the same information to the next resources which lead to the transformation of degraded information. This happens as there are no metrics maintained by the organization at the time of knowledge transfers. This improper information or incomplete information gives chance to develop with low quality in further development process. Improper training over the Integrated Development Environments, other software development, and deployment tools and poor training on external and internal environments delays the development process. This decreases the time of unit testing and functional testing resulting in delivery of low quality software.

Allotting a new resources to maintenance project

A new resource cannot ask all the documents that he need for complete flow and execution of the task as he is unaware of the communications that are performed with client and other Technical persons. This impacts not only to increase in the time of coding and development, but also decreases the quality of the software developed.

Increases training cycle

As the new person comes in to the new project always keeps enquiring about the flow of the existing project and functionality over the existing bugs. Time will be elapsed always reading the existing documents. Reading the communications with the client, communications among the team. New installation process, new set up of environments, communication new documents. From the projects perspective it consumes more time on training on it to continuously newly added resource.

Change of Domain

A person who may be expert in one domain, but when comes to new domain again he is a new learner in that field. Basing on the need of the person or any other background reasons willingly or unwillingly he enters into the new domain. Experiences new bouncers with new terminology, new technology, new platform, and new domain knowledge.

A newly entered person in to a new domain always in a position to accepts what communications reaches to him not in position to strongly clarify his doubts until he understands the basic knowledge over the new environment. This leads to wrong estimations for development and QA cycle as well as maintenance.

Maintenance of Documents and other resources:

A document always speaks and represents more efficient accurate communication and keeps alive of the passed and events with the client and the software company who develops the project for them. It gives true representation and exists as witness to the all committed and base lined requirement and working functionality. The proper scope and meaning will be assessed by a base lined document.

A client not interested in maintaining all the business and technical documents that happens to be realized in all the communications. Also an organization does not keep any person as key responsible to maintain all business and technical documents. Also there will be no proper version control of all the communicated documents. The new team member will be shared by the existing documents that are given by their current superiors. This gives a chance of missing of some documents which gives raise a gap of technical and business communication to the new member. This leads to develop a less quality software for further releases.

A software release always is a combination of core product developed by the project and product team along with the set of client specific requirements which may not be supported by the product team. A patch work will be performed in extra to the original core work to include all client requirements. In this context its be comes important to have product documents and project documents and the patch work documents.

Chances of committing mistakes form client side happens when they lost the historical information of the project or missing of some important documents or unavailable of some information. Due the incomplete perspective of project or module they may raise fake bugs that creates false alarm for investigating of fake bugs.

Maintenance of communications

An organization leaves minutes of meeting, maintenance and communication information responsibility on the persons who are involved at that time process executed does not keep high importance on what has communicated and how it was done. Their maintenance will depend on the concerned person interest, intelligence and level of their role and responsibility. As the project grows from basic level to higher the importance of the documents increases as they go on updation along with the client requirements.

The priority documents like Client estimated documents and Client reviewed functional documents gives the magnitude of the project and expresses the complexity of the project. The requirement documents keeps on changing as the client's needs increases or requirements changes , if version controlling not maintained properly , and if it not communicated , there is chances of the overlooking some of minor requirements which impacts over all quality.

In general a project manager or delivery head and Business Analyst and Technical lead and a quality control manager will involve in conducting the client meetings. Each person will have different perspectives over the project. Every body expresses their view of solution about a particular problem orally while discussing and leaves the responsibility of the document to either business analyst or other, who prepares the document in their perspective. In such scenario the other perspectives will be missed. The representation and assessment of whole problem will be narrowed towards specific problem. For example they may miss managerial, technical and quality oriented information. As the days passes

every body lost focus on it. This happens as a general practice which leads to loss of quality issues by increasing the time of discussion of old issues reviewing old poor perspective documents.

Dependency with Core-Product team:

An issue which is reported by the client can be fixed only by the maintenance team if it is in the control of them. Most of the issues can be analyzed and debugged as per the logs and scenarios and test cases that are submitted by the client. An issue will be resolved basing on the available information and investigations that are performed at various levels of application and environment. If the issue is in code base of project can be fixed very easily but if it is an environment issue that may or may not be fixed depends on the artifacts available. If not they should be arranged by the product team. If the issue is not in the control of the project then it should be escalated to the product team.

Product team will respond to the issues basing on the severity of the issue. Also they can provide he issue at when they have successive releases of the related product. Most of the time the urgent issue can be addressed by the product team by giving a new hot fix to the project team. The project team again has to be educated about the behavior and functionality of the hot fix that should solve the bug of the client. Some times the hot fix may introduce new bugs apart from solving the old issue. In that scenario the existing code and process to build the project has to be re engineered to avoid unexpected results and correct functionality.

Most of the time product team is interested in building generic application that meets the common requirements of the projects of various clients based on the different client requirements. Also the product team has their own software development life cycle .Most of the time product should has an extension point , so that any project that may need a different feature which is not supported by the product can be build by using it. So that it can use all the features that are provided by the product optimally. If the product does not have the extension point at the situation where the client bug is occurring then the project team has to do reach and development and has to find out a by pass or new tool which can fix that issue with in a limit amount of the time. Because most of the bugs should be delivered in the consequent builds of the previous builds as they impact seriously on the client base and impact on the reliability of the product (security and usability issues).

Continuously changing the Project Managers:

Effects the area of non baseline documents.

When a project is under the state of development, if a Project Manager is changed, it will have an impact on the commitment of the previous requirements. The former Project Manager has estimated the requirements depend on his personal experience merged with existing environmental factors and past communications with client and the knowledge of client base. Normally the Project Manager or Business Analyst are sent to the client base to understand and analyze the vague requirements primarily then the Project Manager and Business Analyst will acquire knowledge on the project and when the project development reaches to a logical end the management move the Project Manager to new project. When new Project Manager enters into the project, the view of new Manager may not synchronize with former Manager. It gives raise to new estimations for a non base lined document.

Effects work allocation and Team management

People always try to compare with a new person with old person. New terminology, new approach with old. Tries to feel happy with old and past manager and tries to find mistakes with new one. Accepting the new person comes to pictures when they find the good and likely quality with them. A new manager always tries to think to get the grip over the team to run the show smoothly and always tries to find the mistakes with the team to control the team member.

A new manager will definitely wont have grip over the client, also the client may put more nonsense with a new persons to show their highness, also expects more explanations form the manager over the issues that come across the build releases. A new manages has to understand the mentality of various clients and languages and touch points and designations of the client. The nonsense communication stops only when the client gains belief and Communications towards the team Need to train on point of contact, need to follow the new process that is implemented by the new Project Manager.

Miscellaneous factors:

Lack of proper Infrastructure

Software development should be performed with proper environment like good configuration hardware systems and access to the internet site and proper training to the newly coming cutting edge technologies along with supporting and aid of other documenting tools and modular deployment tools. A developed software need to be tested thoroughly to give a good quality. Testing process not only includes the human testers with good experience on the particular domain but also includes the testing environments like simulators, Electronic devices. These electronic devices are always found to be more expensive. They will be shared with all the team

members and between the projects basing on their availability. Due to the lack of these devices or testing on the older version devices leads to poor quality. Because some of the issues can be identified only on latest versioned devices.

Software releases and Up gradations related issues

Software upgradations are very common in software development life cycle process to build a robust and stabilized product. Every software company releases the product versions as the product grows. New functionality is added to the existing product along with bug fixes in the previous versions of the product. Along with the software releases they has to release help manuals ,user manuals, trouble shooting manuals, Instructions manual user guide manuals, Technical manuals, product manual, help documentation Reference manuals related to that particular version. These documents should carry all the information regarding the previous and present and future status of the product services. Generally software development companies do not give much importance for verification, validation and review of these documents which causes the low quality development of the projects.

Upgradations of the software product or Integrated Development Environments needs adding list of new software artifacts , programs and resources and new documents. While upgrading new version up gradation process has to keep track of all existing features and functionality so that upgradations should not lead to new bugs and errors. Most of the times they may add new features but ignore some of the features in exceptional conditions related to existing project.. The product has to provide extension point for the new features that are included so that it will fit to different projects.

Environmental issues

Normally Development and testing environment are small scale devices. These consist of software of hardware equipments with lower configuration or simulators. Most of the time they are not upgraded to the cutting edge technologies and do not meet client environment requirements as they found to be very expensive. Some times they may be available and some times they may not be available completely throughout the testing process for required testing team. Testing the application on poor environment does not give complete quality. Most of the post production issues arise due to the lack of proper environment facilities.

The software test bed or development environment could consist of a client server application, Relational Database Management System (RDBMS), middleware, interfaces, daemons, customized processes (written in any software programming language), FTP utilities etc. Functional test phases such as Unit, Integration, Acceptance, all manner of performance or non functional testing and development phases all require IT Environments. The primary clients of an IT Environments Management Function are Software Project and Test teams. The application behavior can be tested only in the production environment to identify all sorts of quality issues.

IV CONCLUSIONS

Allocation of human resources should be of same or related domain and shifting of resources from one project to another should have minor impact on existing project growth and should follow some metrics for allocating them. Metrics should be created for Knowledge Transfer sessions to retain quality of exiting software product and should verify that a new resource person should gain required knowledge related to specific project before allocating to maintenance project also the process of knowledge transfer and allocation should be metric.

Version Controls and resource control software should be used in maintenance of communications. They should be kept for accessible to all resources. Dependency with Core-Product team should be very less. This can be achieved by extension points at various levels along with backward compatibility.

Project Managers should be kept constant till at least one life cycle of the project to have proper vision and proper growth for project to avoid poor software quality. Minimum metrics should be followed for arrangement of proper Infrastructure and Software Up gradations and internal and external Environments. IT Environments Management have to encompasses a set of best practices proposed to provide an effective, end to end management service for test software platforms or development environments.

REFERENCES:

1. http://www.pearsonhighered.com/samplechapter/0201729156.pdf

2. http://www.borland.com/resources/en/pdf/solutions/lqm_driving_quality.pdf

3.http://www.infosys.com/IT-services/independent-validation-testing-services/white-papers/Documents/operational-excellence.pdf

4. http://research.microsoft.com/apps/pubs/default.aspx?id=70535

5. http://research.microsoft.com/pubs/70535/tr-2008-11.pdf

6.http://projectmanagement.ittoolbox.com/groups/strategy-planning/projectlifecycle-projectmanagement/environment-management-3186627

7. http://www.iiitd.edu.in/~jalote/papers/UseMetrics.pdf