Software Engineering for Practiced Software Enhancement

Rashmi Yadav[1], Ravindra Patel[2] and Abhay Kothari[3]

**[1] RGPV ,CSE,Rajiv RGPV Bhopal Transnational Knowledge Society Group of Institutions
Indore, Madhya Pradesh , INDIA**

**[2] RGPV ,Computer Application ,Rajiv Gandhi Proyodigiki Vishwavidalaya
Bhopal, Madhya Pradesh , INDIA**

**[3] RGPV ,CSE,RGPV Bhopal Indore Institute of science & Technology-II
Indore, Madhya Pradesh , INDIA**

Abstract

Software development scenario particularly in IT industries is very competitive and demands for development with minimum resources. Software development started and prevailed up to an extent in industry without the use of software engineering practices, which was perceived as an overhead. This approach causes over use of resources, such as money, man-hours, hardware components. This paper attempts to present the causes of inefficiencies in an almost exhaustive way. Further, an attempt has been made to elaborate the software engineering methods as remedies against the listed causes of inefficiencies of development.

Keywords: *Author Guide, Article, Camera-Ready Format, Paper Specifications, Paper Submission.*

1. Introduction

Software engineering encompasses three major aspects of software system development. These are [1]--

a. System Development Life Cycle
   This consists of four fundamental phases termed as analysis, design, coding, and testing.
b. Software Quality Assurance
   This involves activities such as, audits, reviews, measurements, inspections and walkthroughs.
c. Software Project Management

This involves project planning, monitoring and control, human resource management, As per software development guided by software engineering practices 10-15% time of the total development process should go for coding; 30-40% time should go for analysis and design phases; and 30-40% time should go for testing [1]. In development scenarios where software engineering practices are not followed the process roughly involves understanding the problem and coding it to implement the solution and test the program in minimum quantum so as to make it work. This is a very unsystematic way of working which could lead to omission of various errors left hidden in software, which would reveal later in usage, which, are very expensive to remove, since this involves lots of backtracking (traveling backwards) and reworks while fixing the errors. Software development without software engineering is like making building without map and a consultant/engineer.

Software engineering is nothing but to incorporate engineering approach of problem solving in building any software so as to create it efficiently (using minimum resources, mostly within time and funds), and with quality. When we say 'Engineering approach of problem solving,' we imply first to understand the problem (Analysis), then to find the methods of solution (Designing), further to implement these solutions (Construction), and lastly to test the realized solutions (Testing). This common engineering philosophy applies equally to any of the engineering disciplines of mechanical, electrical, electronics, and others. Up till now we have discussed the meaning and application of engineering approach to software domain, and clamed that all this leads to efficient development of the software. The next section makes the elaboration of the issues of inefficiencies in software development. Subsequently, we discuss the role of software engineering practices in overcoming these inefficiencies.

Apart from planning and monitoring the project well which are more of non-core or support activities under software engineering practice, the core activities like, analysis, design, coding and testing are also prescribed with methods of conducting them efficiently under the subject of software engineering. And, these methods for each activity and their sub-activities have been prescribed by researchers, scientists and practitioners of this field, which again are the outcomes of their prolonged and sincere experience. These methods are empirical and logical ways of conducting the developmental activities in and efficient and effective way. Without the support of these methods it is very difficult to handle the various attributes or cost drivers of any product development scenario. These attributes have been categorized as product attributes, computer machine attributes, personnel attributes, and project attributes. In other words if we try to handle these attributes termed as cost drivers without the support of software engineering methods and tools, the cost or inefficiency of software project development is going to go higher. These cost drivers are listed and defined further—

Product Attributes [2]:

Reliability	It is the probability of a system to work for T hours even in the presence of faults.
Portability	It is the feature that a software can work on more than one hardware and operating system combinations.
Efficiency	It is the extent to which program while execution requires computational resources such as, main memory and CPU time.
Usability	East of learning, understanding and operating a system
Functionality	How well a software product provides functions to cater to the user needs
Maintainability	How easily the software can be changed

Computer Attributes [3,4]:

Execution Time Constraint	CPU is available for t seconds only so the programs should execute in this much time only
Main Memory Constraint	Executable program should occupy only M bytes as these many are only available
Virtual Machine Volatility	Operating system changes its features over the development duration

Personnel Attributes:

Analyst Capability	It is how experienced is system analyst is, if his experience is lower the analysis would not be good
Application Experience	It is the experience of the developers in a particular domain such as, banking, insurance, accounting, automobiles, etc[5].
Programmer Capability	How logical and matured the programmer is ? It is related to the creativity and productivity index of the programmer.
Virtual Machine Experience	It is the experience of the programmer with the operating system
Programming Language Experience	It is the experience of the programmer with the language of development

Project Attributes:

Development Time Constraint	It is the development time constraint on project which requires to deploy extra people and software tools and use of special software process models to achieve this. All these efforts increase the cost if not done properly.
Changing Requirements	If the environment of the business organization for which software is being developed is dynamic over time, the requirements may change frequently causing logical errors and rework, leading to cost increase.

Being Efficient with Software Engineering Practices

As far as different drivers of cost are concerned there are no methods other than trail and error in the case of development without software engineering practices is concerned. But, these trial and error way of developing software is full of uncertainty and can introduce undesirable features like, delay, cost overrun, error introduction, rework, effort wastage, discarding work product , etc. All these are unprofessional outcomes leading to overall inefficiency in the development scenario. Whereas, on the other side for each cost driver software engineering methods can be recommended, which will incorporate or handle the attributes in ways such that the resource requirements are reduced to minimum. In this research work attempt has been made to prescribe most efficient methods to manage various attribute of cost enhancements. These are elaborated further--

Product Attributes:

Reliability	Redundancy in the design of hardware and software. Techniques of redundancy are static and dynamic. Under static we have tri modular redundancy and triplicate tri modular redundancy. Under dynamic we have techniques such as hot standby and cold standby
Portability	Avoiding operating systems commands while programming, that means using the computer language syntax only
Efficiency	Good procedural design through algorithm identification/design
Usability	Appropriate use of 4GL techniques/front end tools to design forms and screens. Use

	of good prototypes.
Functionality	Good modularization with modules having functional cohesion, this shall require good requirements analysis; good data design through normalization
Maintainability	Modular design, with modules having high cohesion and low coupling, such that each module has limited complexity; good data design

Computer Attributes:

Execution Time Constraint	Use of good procedural design; algorithm step optimization
Main Memory Constraint	Code optimization through optimal variable selection
Virtual Machine Volatility	Use of good configuration management techniques

Personnel Attributes:

Analyst Capability	If analyst is new, use of standard SRS documentation can help avoid mistakes
Application Experience	Existing code can be reused or prototyping model can be used for software development
Programmer Capability	Documentation on good programming practices as prescribed by ISO, SEI-CMM can help
Virtual Machine Experience	If this is less cost may go high. Reusing existing code, and use of good design patterns can help
Programming Language Experience	Code reuse; use of 4GL tools can help

Project Attributes:

Development Time Constraint	4GL model; RAD model; Incremental model; agile development models; use of CASE tools; proper documentation; time estimation through models like COCOMO model
Changing Requirements	Use of configuration management techniques can help; use of spiral model.

Conclusions

Software development is happening all across the globe in the premises of IT industry units. Challenges are global customer, quality orientation, and cost and duration competitiveness. These expectations can not be met with software development without software engineering practices because the development becomes inefficient here. Software engineering prescribes methods for analysis, designing, coding and testing and also for the various quality achieving and project monitoring and tracking activities which are more or less categorized as umbrella activities. Causes of inefficiencies fall mainly in four categories they are product attributes, computer attributes, personnel attributes and project attributes. Techniques such as proper software process model, code optimization, algorithm design, data design, configuration management, modularization,

coupling reduction, static and dynamic redundancies can help handling different attributes as elaborated.

References

[1] Pressman R.S., “Software Engineering: Practitioners Approach,” 6th Ed., 2005
[2] Agarwal B.B., Tayal S.P, “Software Engineering,” University Science Press, New Delhi, 2008.
[3] Jalote P., “An Integrated Approach to Software Engineering,” Narosa Publication, 1991
[4] Boehm B. “Software Engineering Economics,” Prentice Hall, 1981
[5] Scacchi W., Process Models in Software Engineering,” Institute of Software Research, UCAL Irvine, 2001

First Author I am Rashmi Yadav PhD Perusing I had completed M.Tech in year 2006 & B.E in 2003 presently ia m working in Transnational Knowledge Society Group of Institutions Acropolis-II Indore. Previously I worked with Sanghvi Institute of Management And Science Indore I am associated with CSI India. I had publish 4International & 8 National Paper. I had publish 2 paper in International Journal.

Second Author Dr. Ravindra Patel completed PhD in computer science currently working in Rajiv a Gandhi Proyodigiki Vishwavidalaya Bhopal as a Associate Professor in computer Application previously he was working in SATI Vidisha he had completed phd. In computer Science.

Third Author Dr. Abhay Kothari presently working in Indore Institute of Science & Technology-II Indore. Previously he had worked in Sanghvi Institute of Management & science he had done B.E. M. S. & Phd in computer Science & Engineering.