Research Article

iMedPub Journals
http://www.imedpub.com

American Journal of Computer Science and Information Technology

2021
Vol. 9 No. 10: 114

# Software Project Planning, People Management, and Effects on Product Quality

Obi Ibeto[1*], Mutairu Gbadegesin[2], Imisi Fakunle[3] and Ajayi S Wunmi[4]

[1]Microsoft Cloud Solutions Provider, Egnyte Cloud Solutions Provider, Helping Companies leverage on the Cloud to reach their full potential. Birmingham, England, United Kingdom
[2]Department of Biochemistry, Basic Medical sciences, Public University in Ibadan, Ibadan, Nigeria
[3]Adeyemi College of education Ondo, Ondo, Nigeria
[4]Medical Laboratory, National Veterinary Research Institute, Vom, Nigeria

**Corresponding author:**
Obi Ibeto, Microsoft Cloud Solutions Provider, Egnyte Cloud Solutions Provider, Helping Companies leverage on the Cloud to reach their full potential, Birmingham, England, United Kingdom

E-mail: obiibeto@gmail.com

**Citation:** Ibeto O, Gbadegesin M, Fakunle I, Wunmi AS (2021) Software Project Planning, People Management, and effects on Product Quality. Am J Compt Sci InformTechnol Vol.9 No.10: 114.

## Abstract

Quite recently, enormous efforts have been made to improve the quality of software being produced. One of such ways has to do with the project planning process; another way is to effectively manage the people responsible for the overall success of the software being produced. Several software project plans exist depending on the type or projects and organization. The means of measuring how software is designed and how it conforms to that design is called software quality. Some of the factors that are observed during software quality are scalability, product quality, completeness, correctness, and total absence of bugs. These applied along with effective people management have been used in the past to prevent risk and enhance both delivery time and product quality. However, some gaps were identified in the earlier works done in this area and in project plans and people management designed for evaluating and controlling product quality prompting the development of modern techniques. Hence, this work tries to investigate different type of project plans and people management techniques leaning on the gaps in research; it attempts to create a framework for better software project planning and alleviation with the aim of enhancing delivery time and product quality.

**Keywords:** Software planning; People management; Software project failure; Software project management

**Received:** October 08, 2021; **Accepted:** October 22, 2021; **Published:** October 29, 2021

## Introduction

High product quality is achieved by proper planning rather than by reacting to problems after they are identified. Standards are chosen and processes are put in place to achieve those standards. It is not enough to make sure you get a software project done on time and under budget. You need to be sure you make the right product to suit your stakeholders' needs. Product Quality means making sure that you build what you said you would and that you do it as efficiently as possible. And that means trying not to make too many mistakes and always keeping your project working toward the goal of creating the right product as agreed with the stakeholders.

Project success is measured as the ability to complete the project according to desired specifications, and within the specified budget and the promised time schedule, while keeping the customer and stakeholders happy. For proper project completion both planning and execution need to be properly implemented. Control is used as the monitoring mechanism to ensure that each of the two phases is properly implemented, corrective actions being introduced where there are undesired discrepancies between the project's plan and its execution.

The Application of software development cycles and models restrain the project to certain inalienable truths first of which is that the software project must move from one phase to another; the second is that agreed milestones must be achieved while engaging and updating stakeholders on a regular basis. These two truths form the basis of every successful software project. An example of software development model widely adopted is the agile form of project delivery which has gain tremendous popularity over the waterfall model as far as software project delivery is concerned in North America and Europe. Metrics must also be considered to reduce errors and faults and to also properly assess software product quality. Software metrics also help find defects before they occur, and it also help the project team benchmark the project against global best practices in software engineering while accounting for Time, Cost, and human resources. These metrics can be applied to each software development phase from requirement gathering to final software testing.

In the context of software project management business requirements are incorporated alongside time and budget constraints. The diagram below shows an intricate relationship and how any of these three factors can have an adverse effect on the others **(Figure 1)**.


**Figure 1** **Areas of responsibility.**

## Techniques areas of responsibility

Software development and maintenance projects teams do not consist of people working independently or without coordination. Instead, team members are organized in ways that enhance the swift completion of quality products. The choice of an appropriate structure for any project depends on several things (Software Engineering Theory and Practice, 4th Edition by Pflieger and Atlee) [1].

a) The background and work style of the team members

b) The number of people on the team

c) The management style of the customers and developers

During project organization three areas of capabilities and responsibility come to mind and they are:

1) Project leadership

2) Project team,

3) Project board.

The project leadership is saddled with the overall management of the project, project team executes the project while the project board is the lead body that orchestrates project success and decides whether a project must continue or be cancelled [2].

Software project planning is an art. It is a sub-category of project management in which software projects are controlled, monitored, implemented, and planned **(Figure 2)** [3].

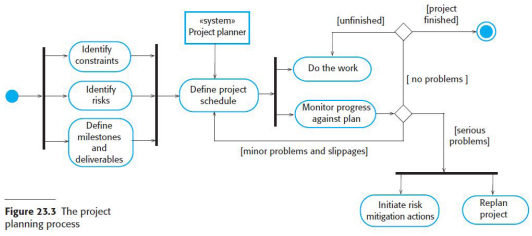


**Figure 2** **The Project planning process.**

The goal for all software project managers is to bring a software project to completion on time, within the budgeted costs, and to meet the planned performance or end-product goals by orchestrating all resources assigned to the project effectively and efficiently.

As a result, managers of software development projects increasingly recognize the importance of estimation and planning [4].

However, although they have many sophisticated tools at their disposal, many systems are still delivered way behind schedule, cost far more to produce than original budget estimates, and fail to meet user requirements [5].

It has been reported that, on average, software systems are delivered a year behind schedule, only 1% of software projects finish on time and to budget, and more important, 25% of all software-intensive projects never finish at all [6].

Some examples of where improper planning and inefficient people management has led to either delay in delivery, poor quality or total failure of projects include: NNPC's Human Resource system built on SAP which was proposed to make human resource management efficient but failed due to internal government bureaucracy or some unforeseen factors and this occurred after spending over $30 M.

Another example is the NigComSat-1, a Nigerian satellite ordered and built in China in 2004 was Nigeria's second satellite and African first communication satellite. It failed in orbit after losing power because of abnormalities in its solar array. This was a clear case of improper planning and poor software design in its satellite monitoring software.

These few examples above are just some examples of notable projects that have either failed or did not complete as scheduled due to poor oversight procedure and bad planning.

Here in this work, an attempt would be made to identify proper software planning methods/techniques and effective people management methods with an aim to enhance delivery time and product quality. Since this work tries to address software planning techniques and people management methods, it is deemed fit to introduce its major concepts.

## Quality software planning

Quality software planning alludes explicitly to the activities of the venture supervisory crew as well as the venture supervisory crew pioneer to take part in the activity of setting up and leading an interaction for the reasons for recognizing and deciding precisely which norms of value are indeed pertinent to the product project in general, and furthermore in making a viable assurance with respect to how to fulfill them **(Figure 3)**.

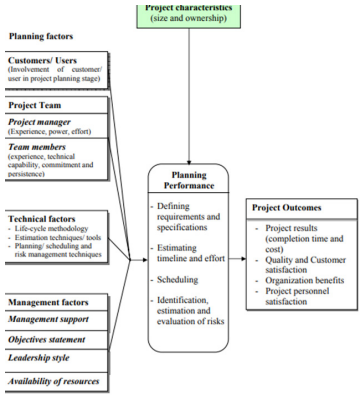


**Figure 3** **Quality software planning.**

Quality planning is typically best done at the onset of the project, but can of course be tweaked as necessary, This term is defined in the 3rd edition of the PMBOK (project management book of knowledge) but not in the 4th.

The motivation behind software project arranging is to recognize the extent of the task, gauge the work in question, and make a venture plan. Undertaking arranging starts with necessities that characterize the product to be created. The undertaking plan is then evolved to portray the assignments that will finish promptly.

The jobs of those answerable for accomplishing the goals likewise ought to be illustrated in the arrangement. For a private company, this may be one individual or key faculty from all through the association, from quality confirmation chief to software testing manager.

The arrangement should set out the lines of power, correspondence necessities and staff levels, abilities, and preparing needs needed to keep up with quality guidelines. In a composed arrangement, set out the timetable of reviews, review work force, and documentation and revealing prerequisite.

Software Quality control and affirmation exercises measure the ideal quality rules for deliverable against existing conditions. This may incorporate mechanical detailing just as day-by-day event logs from quality control faculty. The objective is to keep up with ideal degrees of execution by checking execution of deliverable. Instances of value the board instruments incorporate Zero Defects and ISO 9000; philosophies intended to gauge and address non conformances.

## People management

Hall works on "Correspondence: The Neglected Technical Expertise?" clarified that without a dynamic and imparting human group in a project the likelihood of accomplishment is extremely low. This definition stresses the importance of a team in the context of a software development project.

In our own opinion a team is viewed as a set of people, driven by the same goals of interdependent tasks and capable of taking collective responsibility for the result of their actions within the norms of an organizational culture and structure, Effective team formation in a multi-cultural information technology project environment; the challenges and solutions **(Figure 4)** [7,8].


**Figure 4** **Software development resources.**

• People management techniques in the context of software product quality exist, they are:

• A) Allow software developers do their jobs

• B) Handle Non-development work

• C) Listen and Respond

• D) Encourage Progress

• E) Emphasize Quality over Quantity

• F) Review the Right Metrix

•G) Avoid Task Switching

Now it is clear that software project effectiveness improves from its respective team members [9]. It is people management that contributes either to the software project success or failure. Among software development resources presented as a pyramid, the human resources sit at the top of the pyramid [10].

People management, also known as human resource management (HRM), encompasses the tasks of recruitment, management, and providing ongoing support and direction for the employees of an organization. These tasks can include the following:

a) Training

b) Administration

c) Communication

d) Performance Management

e) Hiring

f) Employee Motivation

g) Safety

h) Compensation

i) Wellness

Brewer [11] gave an analysis of skills and characteristics that must be possessed by a named project or team leader. From experience, we think fundamentally, a project or team leader must have vision and should be able to establish them. She/he must be able to manage conflict and create mutual respect amongst team members, He/she must be able to strike a balance between equity and hierarchy amongst team members and finally, he/she should always give honest feedback. (Effective Team Formation in a multi-cultural information technology project environment; the challenges and solutions [12].

The team/people working to deliver a software development project are the greatest assets (Software Engineering, 10th Edition by Sommerville [13]. It is important that software project managers understand the technical issues that influence the work of software development. As a project manager, you should be aware of the potential problem of people management and should try to develop people management skills.

There are four (4) critical factors that influence the relationship between a manager and the people that he or she manages Software Engineering, 10th Edition by Sommerville.

1) Consistency

2) Respect

3) Inclusion

4) Honesty

Also, to further buttress this point three (3) factors that have the biggest effect on team working are:

1) The people in the group

2) The way the group is organized.

3) Technical and managerial communications.

a) Common causes of software project failure

As the industry has matured, analysis of software project management failures has shown that the following are the most common causes (Why Software Fails", in IEEE Spectrum [14]:

1. Insufficient end-user involvement

2. Helpless correspondence among clients, engineers, clients, and venture chiefs

3.Ridiculous or unstated venture objectives

4.Mistaken assessments of required assets

5.Seriously characterized or inadequate framework necessities and details.

6.Helpless revealing of the venture's status

7.Poorly managed risks

8.Use of obsolete technology.

9.Inability to handle complexity.

10.Sloppy development practices

11.Stakeholder politics

The initial five things in the list above show the troubles articulating the necessities of the customer so that appropriate assets can convey the legitimate task objectives. Another interesting perspective on why software projects fail is because of repetition of task, non-consideration of risk, using wrong process models and customer involvement (Software Development Top Models, Risk Control and Effects on Product Quality by Ajayi et al. [15-36].

**Aims and objectives:** The aims of this work are to examine the possibility of improving software quality through better software project planning and people management to enhance quality product.

The basic objectives are to:

• To evaluate the impact of the project manager on the quality of software project planning and to determine ways of increasing the effectiveness of his intervention.

• Analyze previous project planning models and existing works to establish gap or new trend in this paradigm.

• Identify the basic parameters that must work together to attain quality product.

## Problem statement

It is very imperative to state first that like every sector; software planning process and people management is characterized by different types of challenges.

Earlier research in this paradigm shows that in most cases, success rates of software projects have been found to be lower than expected; and inability to easily identify effective software project planning techniques and people management have been identified as a major factor contributing to the failure of software development projects.

The software industry is also one of the fastest moving and evolving industries, creating an environment where companies can go under as fast as they started, due to domestic and international competition. Business owners, executives, middle management, and all other employees working in this field are continually pressured to keep up, and project management professionals are under even more pressure to ensure the successful execution of projects.

It is not enough to only know about software project management. Project managers must also keep pace with this fast-moving industry to anticipate possible risk, and other factors like cost, quality and third-party integration that may hinder the chances of a successful project. These factors may apply to many industries but due to the speed at which technology changes, and competition increases the pressure to deliver projects on time, on budget, and with the quality standards expected, managing projects in the software industry can feel like being in a pressure cooker.

Thus, the main goal of this work is to review the existing software project planning, people management along quality assurance, quality control and risk management and related works in areas of software project planning and management. After this, then come up with research gaps and ideals on how to overcome the limitations in existing poor planning and management and enhance quick delivery and better-quality products **(Figure 5)**.

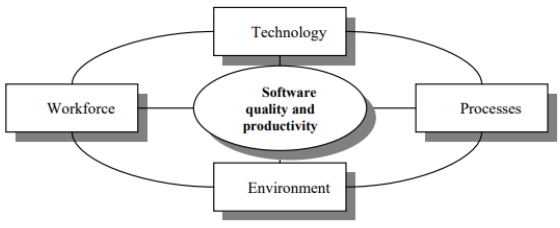


**Figure 5** **Factor that affects the overall Software Quality and Production.**

# Methodology

The methodologies which will be use are:

• Literature search and analysis

• Model adaptation (from generic one)

• Check listing.

## Literature analysis

Software Project Planning is an iterative process because there are many interacting variables, and first assumptions can be proved incorrect [16]. Project managers usually must make early estimates without knowing the detailed requirements.

The project team members and their abilities may also be unknown. However, no one can afford to wait until the end of the Requirements Capture and Analysis (RCA) stage to start planning for the rest of the project.

Software development can be seen as an economic production process [17] whereby inputs (e.g., the effort of systems development professionals) are converted into outputs (systems deliverables), often measured as the size or complexity of the delivered system, using such metrics as Source Lines of Code [18] or Function Points [19].

A large stream of research examines several factors that affect the time required to complete a project and its overall productivity. Studies have shown [20,21] that some of the most important factors that affect the software development process are:

a) Human factors

b) Technical factors

c) Management factors [22].

Nevertheless, studies have pointed out that the project manager's expertise is not captured by the existing models [23]. Instead, software community pays too much attention to the technical factors at the expense of these other contexts. One often cited reason is the difficulty of quantitatively measuring human factors [24].

Nevertheless, the need to improve human resources is quickly emerging as a high priority in the 1990s among IS executives [25].

Information systems development is not considered any more just as a technical process of building an information system, but also as a social process involving stakeholders from multiple organizational units [26].

Successfully building systems, therefore, require effective management of relationships among these stakeholders to elicit their contributions and cooperation, while at the same time, maintaining progress towards the project's goals [27].

Human factors must include factors concerning both developers [28] and users. The degree and effectiveness of participation depends on the relative ability of users and developers to exert influence, their relative power positions and the ability and willingness of each party to communicate Software project management and planning [29].

A model on how users participate in system development is presented by Newman and Robey [30] and a comprehensive review of the latest research on users' participation is given by Cavaye [31].

Nevertheless, the argumentation for user participation has been largely uncorroborated by research evidence plagued, however, with inconclusive and sometimes contradictory results [32].

Nevertheless, previous research investigating the productivity effects of various software engineering management variables involving homogeneous datasets (i.e., data from similar project types at the same firm) have typically reported results with low statistical significance.

For example, Kemerer [33] reported that the addition of the large numbers of productivity factors did little to improve the relationship between size and effort on a set of 15 homogeneous MIS-type projects. Banker et al. [12] also suggested that a relatively small number of critical factors might explain a large amount of the variation in productivity at a specific site. Most of the studies reported in the literature thus far have focused on overall software development. It is suggested [34] that the best approach is the planning of software costs or efforts by phase and activity and the adoption of the bottom-up estimation. Bottom-up estimation is based on estimating the effort for individual tasks, and the effort for the entire project is assumed to be the sum of the effort for each task.

As Kitchenham and de Neumann [35] as well as Lederer and Prasad [36] suggest, to improve standards of cost and effort estimation, it is necessary to adopt an estimation process, based on using different estimation methods and measures at different stages in a project lifecycle and incorporating feedback mechanisms to improve individual estimating expertise and the accuracy of estimation models.

## Methods of evaluating quality in software projects

The literature has identified three leading methods for evaluating planning quality in software development projects:

1) Project management planning quality (PMPQ) model

2) Checklists Model

3) Metrics Mode

**Project management planning quality model:** The quality of software project planning is commonly evaluated through PMPQ Model. This is one of the most effective methods of evaluating quality in software products [37].

In retrospect the Project Management Planning Quality (PMPQ) has two components:

**Project managers know how:** Includes methods or series of steps for which a project manager is directly or indirectly responsible. These methods were obtained from the PMBOK and were arranged together according to its nine knowledge areas.

**Organizational support:** This consists of methods which should be offered by the organization to properly support project methods. These methods were identified mostly from existing maturity models which represent activities that should be performed by the organization. These methods were arranged in relation to the mapping offered by the PMBOK **(Figure 6)**.

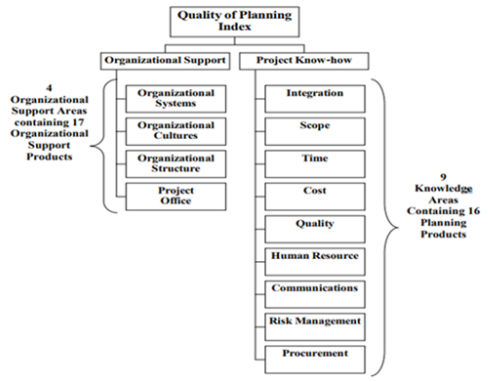


**Figure 6** The PMPQ model structure.

## Checklist approach

Checklists are commonly used during the evaluation after the quality profiles of a software product have been determined. The Evaluation Plan which acts as a starting point contains the quality profile of the software product. The necessary instruments and acceptance criteria are also described in the plan. The evaluation is usually conducted by a testing laboratory or evaluator and the results are documented in a detailed evaluation report **(Figure 7)**.

Also, a checklist contains one or more indicators because they have some degree of importance for assessing quality characteristics for example the indicator modularity of documentation is more relevant to determine changeability than the simplicity of code. A checklist should consist of indicators which relate to one or more quality sub characteristics.

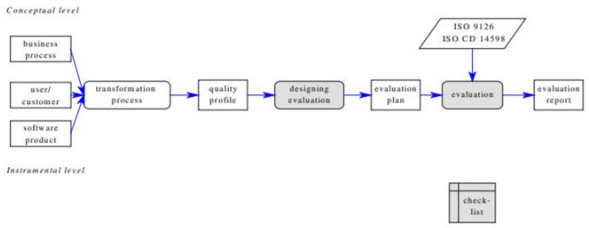


**Figure 7** Software quality evaluation model.

To properly evaluate software product quality, quality sub-characteristics had to be interpreted. Which should ultimately result in direct measures or indicators?

## Metrics approaches

In the context of software quality, Correctness, Absence of bug, scalability, completeness, and product quality are some of the variables that determine software quality. For an organization we cannot quantify the benefits of measuring the software projects from saving time, money to development efforts. Also, when implemented correctly problems can be identified early and resources can be managed prudently. The metric approach to evaluating software projects enable software developers to analyze their code and adjust if required. Metrics can be developed for software testing, software quality, defect analysis, cost estimation and software size [38].

The classification of software metrics can be sub-divided into three units namely:

a) Product Metrics (code metrics)

b) Process metrics

c) Resource metrics

In product metrics the final products such as software code or design documentation is being measured. While in Process metrics the software development lifecycle such as the type of methodology is being measured.

Due to the popularity of Object-oriented programming. The most popular object oriented metrics are Chidamber and Kemerer (CK) metrics [39]. They were designed to measure the object-oriented having features such as cohesion, coupling and inheritance.

In the market there exist numerous software metrics tools; some are open source while others are OEM based. Some of the software metrics tools used in production is listed in the **Table 1** below.

Gaps: After the analysis of the existing works both in software project planning techniques and the careful observations of people management techniques the following were identified as major gaps in their works; From the work of Kris Hughes 2019 a well-organized software project provides the track which to run a successful project; but very little or nothing was mentioned on how project planning handles risk when used in software development and how this could help in product quality.

**Table 1:** Summary of software metrics tools production.

| S. No. | Name | Link to home page | License type | Programming support | Operating system | Supported metrics |
|---|---|---|---|---|---|---|
| 1 | Analyst4j [22,23] | www.codeswat.com | commercial | M | M | WMC, RFC, CBO, DIT, McCable, Halstead effort, volume |
| 2 | CCCC[24] | http://www.sourcerorge net/projects/cccc | Open source | C++,and java file | solaris | Lines of code, McCabe's complexity and metrics |
| 3 | Chidamber and kemerer java metrics[25] | https://www.spinellis.gr/sw/ckjm/ | Open source command-line too] | java | Windows 9x/ME/NT/2000/XP, Unix and Linux | C&K object-oriented metrics |

| | | | | | | |
|---|---|---|---|---|---|---|
| 4 | Dependency finder[21] | https:// depfindsoureforge. net/ | Open source | java | Windows 9x/ME/ NT/2000/XP, Unix and Linux | object-oriented software metrics |
| 5 | Eclipse metrics plug-in1 3 6[26] | http://www. sourcerorge net/ projects.metrics/ | Open source | java | Windows 9x/ME/ NT/2000/XP, Unix and Linux | Cycles in package and type dependencies |
| 6 | Eclipse metrics plug-in1 3 6[27] | https:// eclipsemeritcs. sourceforge.net/ | Open source | java | Windows 9x/ME/ NT/2000/XP, Unix and Linux | metrics during build cycles through the problems view,of range violarions foe each metrics |
| 7 | OOMeter[21,28] | Research papers | commercially available | java,C++,Source code and UML models | Windows 9x/ME/ NT/2000/XP, Unix and Linux | Measure a number of quality attributes (coupling cohesion and complexity and code matrics such as lines of code (LOC) |
| 8 | Semmle[29] | https://semmle com andhttps:// enwikipedia.org/ wiki/semmle | commercially available | object oriented codes | | To search for bugs,measure code metrics |
| 9 | Understand [30] | Research papers | commercially available | java,14 languages,C,C++, | all major operating systems including solaris | Maintaning,measuring and analysing source code |
| 10 | QA-C[5] | Research papers | commercially available | C | Windows solaris (sparc) and RedHat Linux | Halstead metrics and cyclomatric omplexity metrics |
| 11 | Testwell CMT++[5] | Research papers | commercially available | C++ | Windows and Unix only | Complexity measurement tool |

Again, the work treated project planning as a broad topic and did not shed light on the implications when a project is in shambles and the detailed factors of what causes project failure. This brings to bear the well-articulated works of “Software Development Top Models, Risk Control and Effects on Product Quality” by Ajayi. which identifies the reasons why projects fail and mentions the reasons giving succor to the sensitive paradigm.

From the works of Brewer [40] he asserted the qualities, skillset and characteristics possessed by a team leader/project manager but gave little direction as to the overall vision of the project and the importance of the team behind the delivery of the product. This helps shed more light on the position of Effective Team Formation in a multi-cultural information technology project environment; the challenges and solutions, which states the importance of a project leader with an overall project vision, mutual respect amongst team members and striking a balance between equity and hierarchy amongst team members. These two positions in our view are mutually exclusive and give the much-needed interference to delivering product quality.

## Conclusion

Plan-driven development is organized around a complete project plan that defines the project activities, the planned effort, the activity schedule, and who is responsible for each activity ”Software Engineering, 10th Edition by Sommerville”.

People management involves choosing the right people to work on a project and organizing the team and its working environment so that they are as productive as possible. People are motivated by interaction with other people, by the recognition of management and their peers, and by being given opportunities for personal development.

In our view Software development groups should be small and cohesive. The key factors that influence the effectiveness of a group are the people in that group, the way that it is organized, and the communication between group members. Communications within a group are influenced by factors such as the status of group members, the size of the group, the gender composition of the group, personalities, and available communication channels. The success of the product depends largely upon the people initiatives, mental setup of practitioners, logics applied and systematic approach.

In the wider body of knowledge, we believe great work has been done to uncover the mystery behind project planning and people management particularly in software project development, but the recent pandemic COVID-19 pandemic that occurred in the year 2020 has shown that numerous challenges still exist when teams work remotely in a bid to deliver complex software projects. The study and research of these effects will be of immense benefits to humanity.

## References

1 Pflieger M, Atlee (2010) 4th Edition, Software Engineering Theory and Practice.

2 Abran A, Bourque P, Dupuis R, Moore JW (2001) Guide to the software engineering body of knowledge-SWEBOK. IEEE Press 5: 201-202.

3 Armstrong JS (1982) The value of formal planning for strategic decisions: Review of empirical research. Strateg Manag J 3: 197-211.

4 Arnott D, Pervan G (2015) A critical analysis of decision support systems research. In Formulating Research Methods for Information Systems 15: 127-168.

5 Bannerman PL (2008) Risk and risk management in software projects: A reassessment. J Syst Softw 81: 2118-2133.

6 Chatzoglou, PD, Theriou NG, Dimitriadis E, Aggelides V (2000) Software project management and planning: the case of the Greek IT sector. Int J Appl Syst Stud 1: 5–6.

7 Féris MA, Zwikael O, Gregor S (2017) QPLAN: Decision support for evaluating planning quality in software development projects. Decision Support Systems. 96: 92-102.

8 Youll DP. Making software development visible: Effective project control. 1990 Nov 27.

9 Kemerer CF (1987) An empirical validation of software cost estimation models. Communications of the ACM 30: 416-429.

10 Kemerer CF (1993) Reliability of function points measurement: a field experiment. Communications of the ACM. 36: 85-97.

11 Locke EA, Latham GP (1990) A theory of goal setting and task performance. Prentice-Hall.

12 Banker RD, Chang H, Kemerer CF (1994) Evidence on economies of scale in software development. Inf Softw Technol 36: 275-82.

13 Constantine LL (1993) Work organization: Paradigms for project management and organization. Communications of the ACM 36: 35-43.

14 Cusumano MA, Kemerer CF (1990) A quantitative analysis of US and Japanese practice and performance in software development. Management Science. 36: 384-406.

15 Low GC, Jeffery DR (1990) Function points in the estimation and evaluation of the software process. IEEE IEEE Trans Softw Eng 16: 64-71.

16 Lauer, T.W. (1996) Software project managers' risk preferences J Inf Technol 11: 287–295.

17 Nidumolu S (1995) The effect of coordination and uncertainty on software project performance: residual performance risk as an intervening variable. Inf Syst Res 6: 191-219.

18 Mukhopadhyay T, Kekre S (1992) Software effort models for early estimation of process control applications IEEE Trans Softw Eng 18: 915-923.

19 Ali AI, Seiford LM (1993) The mathematical programming approach to efficiency analysis. The measurement of productive efficiency.22: 120-159.

20 Subramanian GH, Zarnich GE (1996) An examination of some software development effort and productivity determinants in ICASE tool projects J Manag Inf Syst 12: 143-160.

21 Perry DE, Staudenmayer NA, Votta LG (1994) People, organizations, and process improvement. IEEE Software. 11: 36-45.

22 Rasch RH, Tosi HL (1992) Factors affecting software developers' performance: An integrated approach. MIS quarterly. 1: 395-413.

23 Markus ML, Bjørn-Andersen N (1987) Power over users: its exercise by system professionals. Communications of the ACM. 30: 498-504.

24 Macala RR, Stuckey LD, Gross DC (1996) Managing domain-specific, product-line development. IEEE Software.13: 57-67.

25 Beath CM, Orlikowski WJ (1994) The contradictory structure of systems development methodologies: Deconstructing the IS-user relationship in information engineering. Inf Syst Res 5: 350-377.

26 Boehm BW (1991) Software risk management: principles and practices. IEEE software.8: 32-41.

27 Boehm BW (1984) Software engineering economics. IEEE Trans. Softw Eng 1: 4-21.

28 Brooks FP (1974) The mythical man-month. Datamation. 20: 44-52.

29 Charettte R (2005) Why software fails, IEEE Spectrum 5: 42-49.

30 Dalcher D and Brodie L (2008) Requirements change management: Why are current change request forms inadequate? In Software Measurement European Forum. 28 101-115.

31 Standish G (1994) Charting the Seas of Information Technology-Chaos. The Standish Group International. 19: 178-183.

32 Linberg KR (1999) Software developer perceptions about software project failure: a case study. J Syst Softw 49: 177-192.

33 Verner J, Sampson J, Cerpa N (2008) What factors lead to software project failure? In 2008 second international conference on research challenges in information science 3: 71-80.

34 Xi'an, alami Lasisi (2018) presentation on nigerian satellite augmentation system (nsas) role/contributions to gnss and request to join gnss providers' forum.

35 Ajayi W, Adekunle YA, Awodele O, Akinsanya AO, Eze MO (2018) Software Development Top Models, Risks Control and Effect on Product Quality. Glob J Comput Sci Inf Technol 12: 134-136.

36 Ajayi sw, Omotosho J (2000) Effective Team Formation in A Multi-Cultural Information Technology Project Environment; the challenges and Solutions.

37 Mishra A, Misra S (2010) People management in software industry: the key to success. ACM SIGSOFT Softw Eng Notes 35: 1-4.

38 Chidamber SR, Kemerer CF (1994) A metrics suite for object oriented design. IEEE Trans Softw Eng 20: 476-493.

39 Rishabh (2013) Software Project Management Paperback. 14: 150-156.

40 Zach H (2016) Managing Software Developers Effectively. 5: 210-217.