MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 2, 2026, Pages 379-382

COMPARATIVE ANALYSIS OF PROPOSED AND EXISTING UNIVERSITY SOFTWARE APPLICATION MODELS

Prof. (Dr.) Sanjay Kumar[1]
[1]Department of Computational Sciences, Brainware University, Kolkata, West Bengal.
Sonu Rana[2]
[2]Assistant Professor, Department of Computer Science and Engineering,
Brainware University, Barasat, Kolkata, West Bengal, Pin 700125, India.
Sreelekha Paul[3]
Assistant Professor, [3]Department of Computer Science and Engineering,
Brainware University, Barasat, Kolkata, West Bengal, Pin 700125, India.
[1]Sanjaysatyam786@gmail.com

ABSTRACT
The field of software engineering is identified with the development of software. Substantial software needs precise development. Basic Programs can be produced without following any orderly methodology. From the most recent couple of years, the PC business has experienced progressive changes in hardware just as in software. Software management is a process that indicates the zone of software development through the process models, which are known as software development life cycle. A short review about various models of software development and examination between them was analyzed. This Paper sorts and determines various strategies for depicting or demonstrating how software frameworks are produced. It starts with foundation and meanings of customary software life cycle models and current software development practices.

Keywords: Software design, software management process, software design, framework activities, software development life cycle.

1. INTRODUCTION

The significance of computer is expanding step by step. Computer has turned out to be well known in various fields, for example, industry, prescription, trade, instruction and even horticulture. Presently a days, associations turn out to be progressively reliant on computer in their fills in because of computer innovation [4]. Computer is viewed as a timesaving gadget and its encouraging aides in executing intricate, since a long time ago, rehashed processes in an exceptionally brief time with a rapid. Notwithstanding utilizing computer for work, individuals use it for entertainment only and excitement. Perceptibly, the quantity of organizations that produce software programs to facilitate works of workplaces, organizations, banks, and so on, has expanded as of late which results in the trouble of identifying such organizations. Today, software goes up against a double job. It is a product and, in the meantime, the vehicle for conveying a product. The greater part of the general population consider the program and software [2] to be same. Be that as it may, software comprises of projects as well as the supporting reports. Software lifecycle models are utilized as apparatuses for arranging and observing software ventures. Diverse models have been proposed. Each model's viability changes with task conditions. It is broadly recognized that no single model is viable in all circumstances. Along these lines, a powerful model must be chosen for each task. A software process model[3] is an abstract portrayal of a process. It introduces a portrayal of a process from some specific point of view as: Requirement Gathering: To create software, first we need to accumulate every one of the prerequisites from client or partner. Arranging: Basing on the necessities, spending plan, time must be acclimated to build up the software. Analysis and Design: Complete Analysation is done on the necessities and arranging. After total investigation designing is begun. For the design, diverse devices are utilized. Usage: For the above design, code is produced with appropriate programming language. Testing and Maintenance: After fulfillment of creating software, testing is performed. On the off chance that the product is very much tried, at that point it will go for alpha and beta tests. The above activities are called casing work activities. In any software life cycle models or software process models these five activities are performed and altered.

2. SOFTWARE APPLICATIONS

Software might be applied in any situation for which a prespecified set of procedural advances (i.e., a calculation) has been characterized (prominent special cases to this standard are master framework software and neural system software). Information substance and determinacy are critical factors in deciding the idea of a software application[4]. Content alludes to the importance and type of incoming and outgoing information. The accompanying software zones demonstrate the expansiveness of potential applications[5]. Framework software.: System software is an accumulation of projects written to service different projects. Some framework software (e.g., compilers, editors, and document management utilities) processes are mind boggling, however determinate, information structures. Different frameworks applications (e.g., working framework segments, drivers, media communications processors) process to a great extent uncertain information. In either case, the framework software territory is described by substantial communication with computer hardware; overwhelming use by different clients; simultaneous activity that requires booking, resource sharing, and modern process management; complex information structures; and various outside interfaces. Constant software: Software that screens/dissects/controls genuine occasions as they happen is called continuous. Components of ongoing software incorporate an information gathering segment that gathers and configurations information from an outside domain, an analysis part that changes information as required by the application, a control/yield segment that reacts to the outer condition, and an observing segment that arranges every other segment with the goal that continuous reaction can be kept up. Business software: Business information processing is the biggest single software application zone. Discrete "frameworks" (e.g., finance, money due/payable, stock) have developed into management information framework (MIS) software that gets to at least one huge databases containing business information. Applications around there rebuild existing information in a way that encourages business tasks or management basic leadership. Notwithstanding regular information processing application, business software applications additionally include intelligent figuring (e.g., pointof-deal transaction processing). Engineering and logical software: Engineering and logical software have been portrayed by "calculating" calculations. Applications go from cosmology to volcanology, from car stress analysis to space transport orbital elements, and from atomic science to computerized manufacturing[7]. Be that as it may, present day applications inside the engineering/logical territory are moving far from customary numerical calculations. Computer-helped design, framework reenactment, and other intelligent applications have started to take on constant and even framework software qualities. Embedded software.:Intelligent products have turned out to be typical in about each customer and mechanical market. Embedded software dwells in read-just memory and is utilized to control products and frameworks for the shopper and modern markets. Embedded software can perform constrained and obscure capacities (e.g., keypad control for a microwave) or give critical capacity and control ability (e.g., computerized capacities in a car, for example, fuel control, dashboard shows, and slowing mechanisms). PC software:The PC software advertise has blossomed in the course of recent decades. Word processing, spreadsheets, computer graphics,multimedia, excitement, database management, individual and business budgetary applications, outer system, and database get to are just a couple of many applications. Online software:The Web pages recovered by a program are software that joins executable guidelines (e.g., CGI, HTML, Perl, or Java), and information (e.g.,hypertext and an assortment of visual and sound arrangements). Fundamentally, the system turns into a monstrous computer giving a practically boundless software resource that can be gotten to by anybody with a modem. Man-made reasoning software: Artificial insight (AI) software makes utilization of nonnumerical calculations to take care of complex issues that are not agreeable to calculation or direct analysis[4]. Master frameworks, likewise called knowledgebased frameworks, design acknowledgment (image and voice), counterfeit neural systems, hypothesis demonstrating, and amusement playing are illustrative of applications inside this class

3. BACKGROUND STUDY

A few number of well known software development approachs have been introduced over the most recent few years, for example, - Agile development is spoke to be a creative and open exertion to address users' needs focused on the essential to convey appropriate working business applications quicker and cheap. The software is typically given in steady or iterative design. The spry development approaches are commonly worried about proceeding with user inclusion through the utilization of design groups and uncommon workspaces. The gave increments are probably going to be minor and deficient to little supply stages to affirm fast end. The association procedure devoured depends on the burden of time boxing, the stringent supply to objective which orders the conceivable outcomes, the accumulation of exhibitions to be given and the changes to experience the objectives. Nimble development is generally advantageous in situations that change progressively and execute challenges of constrained outcomes. Spry methodologies bolster the origination of parallel increment and dissemination inside a by and large purposeful framework [9]. It is technique to increment, in light of the advancement and arrangement of extremely immaterial increases of execution. It trusts on nonstop code upgrade, user commitment in the development group and pair reasonable programming. It very well may be hazardous to keep the enthusiasm of buyers who are associated with the process. Colleagues might be contradictory to the infiltrating support that portrays lithe methodologies. Featuring variances can be dangerous where there are a few partners. Assentions might be a troublesome likewise with different techniques to steady development. [6] A huge number of writers discharge and specialized diaries treat the Scrum as the best strategy to software development. Be that as it may, the first Scrum technique isn't worthy for progressively work in deft condition in a systems administration management. Because of that, analysts protracted the Scrum-based model [8] and the Kanban philosophy to benefit as much as possible from both and locate a progressively adequate strategy for working software development in capably dissipated light-footed foundation.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 2, 2026, Pages 379-382

4. PROBLEM STATEMENT

There are couples of software techniques which are broadly utilized by numerous software development associations. Yet, here and there, associations particularly new organizations face issue to choose the software strategy for a particular software venture. Wrong choice of software development techniques frequently makes those undertaking fizzled. There are a few number of works have been introduced for near investigation of software techniques. Be that as it may, in our investigation we discover a few defects or weaknesses for assessing the software systems. For instance, we locate that numerous basic parameters for models Communication, Requirement Specifications, Cost, Resource Control, Simplicity, Risk Analysis Feedback from User, Customer Priority, Precondition, Elasticity, Practicality, Implementation, Usability and so forth - are not considered in their investigation. So that occasionally their proposal for picking the software development strategies may not compelling in such manner. In addition, their method for estimating the assessing parameters are not right, since they didn't characterize the important estimations of those parameters. Moreover, it isn't make reference to plainly how these estimations of the applicable parameters originate from. In this way, missing parameters are required to be considered for an appropriate similar investigation of the software development approaches.

5. PROPOSAL FOR COMPARING SOFTWARE DEVELOPMENT METHODOLOGIES

In our research, we are going to deliberate about which process to choose for best methodology to deliver the quality software to the customer. Before determining the process to be used, we should get solution for some queries.

1. Who is the ultimate consumer for the software?
2. What is the opportunity of the product?

3. Where are the Development teams placed? Our Development is to discuss for the prerequisite given by the customer which methodology to be used. Let us have a comparative study which process will be active in the below processes and the Advantages & Disadvantages of choosing the model. For this research and comparative studies between different Software Methodologies we have chosen four types of methodologies, which are The Waterfall Model, Scrum, Kanban and XP. We have selected these methodologies, because these four models are regularly followed by several software development industries. In these process models Waterfall is much known to all, and now Agile is very popular globally, and Scrum, Kanban and XP are the family members of Agile Software Development (ASD). Waterfall Model is a further name for the more longestablished approach to software development. It's called 'waterfall' as this category of development is often planned using a Gantt chart – developers complete one segment before touching on to the next segment. In Waterfall, it's rarely aim to return to a segment once it's finished. As corresponding, better gets whatever doing right the first time. This process is exceedingly precarious, often more expensive and usually fewer professional than further Agile methods. Scrum process conveys far less hazard than Waterfall processes. It centers around giving completely tried, self-governing, valuable, little highlights. As comparing, it's differentiating the hazard – in the event that one element goes mistaken, it ought not connect with another trademark. All things considered, designers still arrangement work in cycles and they will in any case discharge toward the finish of emphasis. Kanban refined as a sub constituent of the Toyota Production System. In Kanban the framework is imagined: work is destroyed down into little, detached things and composed on a card which is entranced to an area; the board has particular sections and as the work advances through various stages the card is moved thus. In Kanban the amount of items that can be in advancement at any one time is exactingly blemished. The standard time it takes to outright a thing is followed and enhanced with the goal that the process moves toward becoming as productive and unsurprising as could be allowed. The destruction of abuse is abrogating. Related to shorter emphasess, some other critical things which recognize XP from Scrum are: XP groups take a shot at things in a tyrant priority arrange while a Scrum may not unavoidably handle every thing in priority request once in run. XP groups can get new objects of exertion into emphasis and change out objects of comparing size if the buyer picks on another priority. In states of connections, the duty of the customer in XP is exceptionally parallel to that of the Product Owner in Scrum – in that they help to compose user stories, needs them and are constantly open to designers – regardless of the way that less well unmistakable. Commonly Scrum and XP keep up a day by day stand up meeting. The parameters we have recognized for recognizing these philosophies are:

- Communication
- Requirement Specifications
- Cost
- Resource Control
- Simplicity
- Risk Analysis
- Feedback from User
- Customer Priority
- Precondition
- Elasticity
- Practicality Implementation
- Usability We have selected these parameter sets to optimize a model solution to a target solution.

Standard Agile Performs

- The development team was small: 3 programmers and a QA also playing the Scrum Master character. There was an enthusiastic Product Owner who was a Business Analyst acquainted with the problem to be solved. The sponsor, a Product Manager, was also exceedingly convoluted.
- The development team chooses a 3-weeks Sprint span based on unusualness with Agile in overall, the remark that tasks and stories might be bigger than in other developments and the understanding that development team members could not protection for separately well due to concentration.
- Complete system policy was incremental.
- Functionalities were prioritized and done in priority mandate.
- The Product Owner and end-users were convoluted on a daily basis. Challenges to Standard Agile Performs
- The programmers were extremely focused in their technologies.
- Cross-training was inadequate by a vertical learning curve.
- Automated testing was measured to be unbearable. Our team thought this at inordinate length. At the unit phase, well – what is unit in COBOL? It is a structured programming language. Programs are enormous, massive. Accumulates yield an extended period. It is dangerous to run just portion of the code. Testing is done by moving through a debugger and trifling with capricious principles in recollection. This does not loan itself to computerization. Our team observed into screen footage and repetition, which is conceivable, but the subsequent writings are tremendously friable. In the termination, the competent cost of enquiry was too high for the little project timeline.
- Consuming a small development team did reason some postponements when individuals were absentminded.

What We Got

- Throughout backlog estimate, explanation of a story was short-circuited when it developed deceptive that the business was asking for something that was technically unbearable within the downstream host organism. The story, primarily estimated to be moderately large, was merely released. The Product Owner was not troubled at the loss of the features given the fundamentally unlimited cost of the story even though it had initially been protuberant in the inventive project budget. Cost investments were estimated at approximately 100000 BDT.
- Though planning Sprint 2, another great, high priority functionality vaporized when the Product Owner understood that the work done in Sprint 1 caused in an adequately useful explanation to the box of the floor. The functionality was a real-time update of data from one backend system to a front end on additional system. User testing of the first sprint product presented that batch explains had satisfactory price and were much trusting to gadget. Fairy another 100K BDT saved.
- The development team stroked that these investments would not have been comprehended in an old-style project method. The requirements would have been overheated into a design document and applied as well as likely without any conversation with the Professional.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 2, 2026, Pages 379-382

- Variations in other requirements resulted from nonstop review by the Product Owner and were not a subject for the programmers.
- The project was small, but still accomplished so rapidly compared to pre-Agile potentials that the team stayed together to implement another set of functionalities out of the possibility of the original budget.
- The development team, some of who were incredulous, had pleasurable and appreciated the attainment. The Professional was very satisfied as fine.
- The pre-Agile approximation of work was 2 years gone time in this extremely multi-tasked situation. The Agile project was completed in less than 3 months thanks to an absorbed determination. Time worth of currency, which one?
- Cheers to administrative story trimming by the Product Owner and Sponsor, the product conveyed formerly than even the original agile estimated release date.

- Management had a superior familiarity of status without having status intelligences modestly by attending the daily stand-up meetings. Our Statements So, did Agile work out well for this all-mainframe project? Indeed, it did. Even though it was not full Extreme Programming in methodological performs, using a Scrum approach and agile principles resulted in both previous Return on Investment and minor cost. Oh, and everybody had great doing it – always a respectable signal. A software process model is a streamlined demonstration of a software process, presented from a specific perception. A software process model is an intellectual illustration of a software process. Problems solving in software contain of these activities:
- Finding the problem
- Determining a plot for solution
- Code generating the intended solution

- Testing the authentic program On behalf of big structures, each movement can be enormously composite and procedures and procedures are required to complete it professionally and appropriately. Moreover, each of the basic undertakings itself may be so big that it cannot be controlled in particular phase and must be fragmented into smaller phases. For example, design of a big software structure is always fragmented into several, discrete design segments, initially from a very high level design identifying only the mechanisms in the system to a thorough project where the logic of the mechanisms is identified [7]. The basic undertakings or egments to be performed for developing a software structure are-
- Purpose of System's Requirements
- System Analysis & Design
- Developing or coding of the software
- System Testing as an end user
- System deployment and regular maintenance

Table 1 comparison various process models

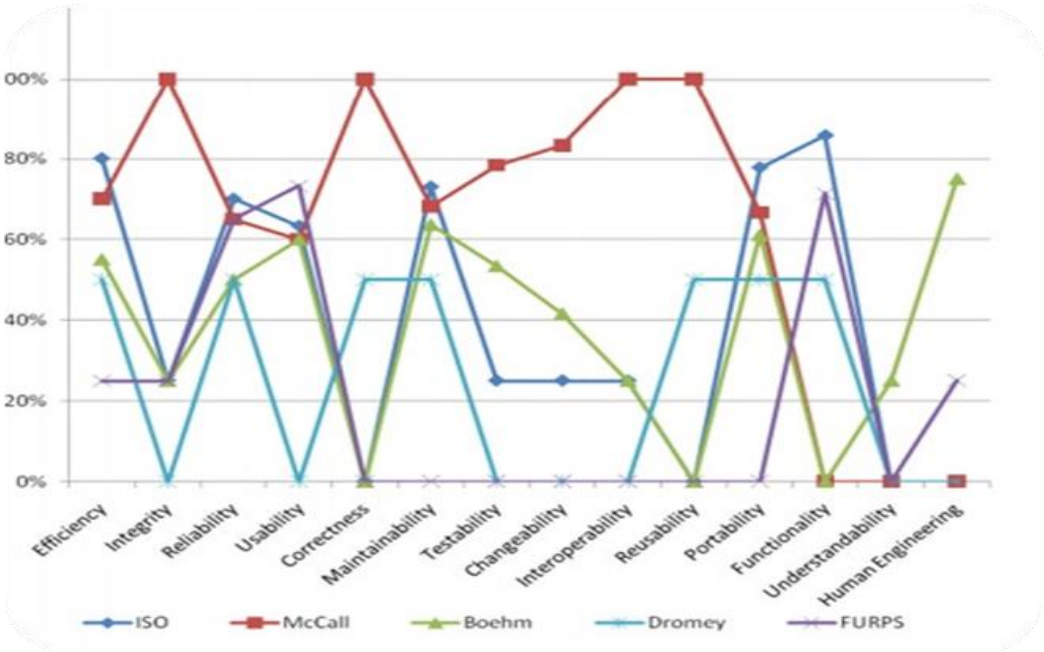
Parameters / Process Models →	Waterfall	Scrum	Kanban	XP
Communication	Initial level	Frequently	Frequently	Initial level
Requirement Specifications	Initial level	Frequently change	Frequently change	Initial level
Cost	Low	Much Expensive	Much Expensive	High
Resource Control	Yes	No	No	Yes
Simplicity	Simple	Complex	Complex	Intermediate
Risk Analysis	Only at beginning	Yes	Yes	Yes
Feedback from User	No	No	No	Yes
Customer Priority	Nil	High	High	Intermediate
Precondition	Requirement clearly defined	No	No	No
Elasticity	No	Very High	Very High	Medium
Practicality Implementation	No	High	High	High
Usability	Basic	Most use now a days	Most use now a days	Medium

This paper examined what software process show is and different process models, likewise contrast them and distinctive parameter and feature the variables for picking them. This paper displays the graph dependent on utilization. Nonetheless, the current model still can be enhancing and changed dependent on less cost, time and high effective. The engineer should discover following viewpoints. The Software life cycle models are the devices that assistance to oversee software life cycle. They will give basic comprehension about life cycles. This paper speaks to correlation of various software process models. These models determine process consistency and enhancement. Each model has its own qualities and shortcomings. A few models are more suitable in certain venture condition than others. The determination of a software lifecycle show for a task is an essential choice. The utilization of an unseemly model can be adverse to extend achievement and software quality. This paper sorts and inspects various techniques for depicting or demonstrating how software frameworks are created , what are the applications of software and so on.. It starts with foundation and meanings of customary software life cycle models and current software development practices.

Table 2. The total value of the software quality models

Factor/Model	*ISO*	*McCall*	*Boehm*	*FURPS*	*Dromey*
Efficiency	80%	70%	55%	25%	50%
Integrity	25%	100%	25%	25%	0%
Reliability	70%	65%	50%	65%	50%
Usability	63%	60%	60%	73%	0%
Correctness	0%	100%	0%	0%	50%
Maintainability	73%	68%	64%	0%	50%
Testability	25%	78%	53%	0%	0%
Changeability	25%	83%	42%	0%	0%
Interoperability	25%	100%	25%	0%	0%
Reusability	0%	100%	0%	0%	50%
Portability	78%	67%	61%	0%	50%
Functionality	86%	0%	0%	71%	50%
Understandability	0%	0%	25%	0%	0%
Human Engineering	0%	0%	75%	25%	0%
Total	**39.27%**	**63.67%**	**38.19%**	**20.34%**	**25.00%**

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 2, 2026, Pages 379-382



CONCLUSION

The Software life cycle models are the instruments that assistance to oversee software life cycle. They will give regular comprehension about life cycles. This paper speaks to examination of various software process models. These models determine process consistency and enhancement. Each model has its very own qualities and shortcomings. A few models are more fitting in certain undertaking condition than others. The choice of a software lifecycle demonstrate for an undertaking is an imperative choice. The utilization of an improper model can be impeding to extend achievement and software quality. This paper sorts and inspects various techniques for portraying or demonstrating how software frameworks are produced , what are the applications of software and so forth.. It starts with foundation and meanings of customary software life cycle models and current software development practices.

REFERENCES

[1] Phadke, D. (2010). Granthalaya Sangnkikaran Aani Aadhunikikaran (4th ed.) Pune: Universal Prakashan.
[2] Raizada, A. S. et al. (1965). Union Catalogue by Digital Computers (Automation indocumentation-1). Annals of Library Science and Documentation, 11(4), 54-76.
[3] Haravu, L. (2009). Emerging Initiatives in Library Management Systems. In International Conference on Academic Libraries (pp. 1-9). Delhi: ICAL.
[4] Murty, D.S.R., & Arora, A.M. (1974). Processing of Union Catalogue of Serials Data Using an IBM system/360 Computer. Annals Library Science and Documentation, 21(3), 88-94.
[5] pandey, S. K., & Sharma. (1995). Fundamentals of Library Automation, New Delhi: Ess Ess Publications, p.131.
[6] 49. Mahapatra, P.K., & Chakrabarti, B. (1997). Redesigning the Library, Delhi: Ess Ess Publications, pp. 1-204.
[7] Haravu, L. (2009). Emerging Initiatives in Library Management Systems. In International Conference on Academic Libraries (pp. 1-9). Delhi: ICAL.
[8] Faisal, S. L., & Surendran, B. (July, 2008). A Report on Automation of Library at Kendriya Vidyalaya Pattom, Thiruvananthapuram: Kendriya Vidyalaya Pattom, P.4.
[9] Choudhari, R. (2010). Managing 21st Century Libraries. In New Dimensions in Library Management (p. 267). Aurangabad: Universal Publication.
[10] Tiwari, P. (2010). Library Automation, New Delhi: A.P.H. Publisher, pp. 10-21.