JOURNAL OF INNOVATION IN COMPUTING | ISSN: 2667-0194

VOL. 1 | ISSUE 1 | JULY 2020 | PAGES 16-26

INTERNATIONAL RESEARCH ACADEMY OF
SCIENCES, ENGINEERING AND ADVANCED TECHNOLOGIES

CRITERIA FOR CHOOSING THE RIGHT SOFTWARE DEVELOPMENT LIFE CYCLE METHOD FOR THE SUCCESS OF SOFTWARE PROJECT

[1] AMAEFULE ANGELA ADANNA *
[2] OGWUELEKA FRANCISCA NONYELUM *

[1,2] *Nigerian Defence Academy, Kaduna, Nigeria*

* *Corresponding author:*
angelbabe4nice@gmail.com
ogwuelekafn@gmail.com

Abstract

This research aims to employ the criteria for choosing a suitable software development life cycle method to achieve success in a software project. The terms life cycle of software development (SDLC), the methodology of software development (SDM), various phases of SDM, models of SDLC, benefits, and drawbacks of models were given. A good research literature review is carried out, a comparative analysis is performed on various models of SDLC. Some criteria that determine the choice of a suitable model for software project success were given. A comparison from different researchers is done using different parameters like an analysis of requirement, development team's status, the participation of the user, type of project, and risk associated based on the various criteria used against different models. The result obtained showed that the best suitable method is the spiral model to be adopted for the success of the development of a software project.

Keywords: *Software Development Life Cycle (SDLC), Software Development, Software, Waterfall Model, Iterative Model, Spiral Model, Requirement Analysis.*

1 | INTRODUCTION

In the olden days, it was observed that programming with a computer system was solely carried out by some experts using some set of instructions. Mostly, the instructions were written using some lines of codes fed into the computer system. These codes were debugged for errors to be executed successfully. Due to the high demand for operations by companies, these resulted in enormous program development. Companies relied on computer systems to perform huge tasks. These programs metamorphosed into a more organized way of executing complicated tasks called software by using the computer system. The standard framework on which software was perceived and established is called a software development life cycle (SDLC). Over the years' researchers have discovered some models of SDLC which can be used by various organization for software projects. According to some researchers, it was discovered that the success of a software project depends on some criteria. In this study, different research works were reviewed and comprehensive details of the different criteria that helped to determine the choice of the SDLC method to be adopted were stated.

1.1 | CONCEPTS

According to A. Farrell [1], software development methodology (SDM) is referred to as a series of procedures taken by an organization to develop a program application. SDM a blueprint used to plan, manage, and control procedures for designing, constructing, and building an information system [2, 3]. According to S. K. Dora and P. Dubey [2], SDM is referred to as SDLC. Also, A. Farrell [1] opined that SDM is a model that is used by an organization to create software considering the steps, tasks, and activities required to develop software projects successfully at a particular time, cost, and resources. In producing software, the following undertakings are essential to plan a project; developing requirements, developing specifications, design of the technical aspect, configuration of the program application, development of the application, running of the application, manual guide of the application, training, support, and servicing.

SDLC is a set of stages that avail the knowledge of how to build software. Every software engineer must have an adequate understanding of the choice of the SDLC model considering the project specifications and the requirements of the business. Consequently, according to demand and requirements, it is advisable to choose the right SDLC model for the success of a business [4]. SDLC is a procedure of designing, building, run by checking and executing a software [5]. SDLC is a process of developing a result of software. It is an organized means of building a typical application from software [6].

According to S. Kaur [7], the author stated the stages of SDM to be gathering of requirements, analysing it, design, actual codes, running of the application, servicing it, and providing help during implementation as shown in Figure 1.

Analysis of requirement is the first phase during SDLC, to understand the problem and document the requirement of users, knowing what exactly should be provided by the system. The result obtained is the software requirement specification (SRS) with full functionality description. The next phase is deigning. It is an innovative phase in SDLC, it is a strategy to resolve the problem. The objective is to convert requirement specification to a strategy, resolving the issues associated with the software. The result is called the Software Design Document (SDD). Coding is the next logical phase. It is where SDD is transformed into a written program and is implemented. Testing is the next phase where the result of the program will be tested if the goal was realized. It is the most vital and influential phase of SDLC where it could be determined if the software developed is of a high-quality standard with minimal cost of repairs and reliable outcome. Maintenance is the last segment where the software application built is deployed to the actual user who is responsible for maintaining the software [2, 7, 8].

Figure 1: Phases of SDLC [7].

Many application program development methods and strategies are utilized at the progress stage of the applications. The methods are called "software development process models". Every of the model adopts a distinctive procedure to achieve a breakthrough in the development of an application. Some of the models are waterfall, spiral, iterative, incremental, prototyping, v-shaped, rapid application development (RAD), agile, rational unified process, and extreme programming (XP) models [2-4, 7, 9-12]. A comparison was made on the advantages and disadvantages of some models in Table 1 and Table 2 [3].

Table 1: Comparing the advantages of different models [3].

S/N	Waterfall	Iterative Model	Spiral Model	V-Shaped Model
1.	Simple and easy to use	More flexible than the basic waterfall model	High amount of risk analysis	Simple and easy to use
2.	Easy to manage due to the rigidity of the model – each phase has specific deliverables and review process	If there is personnel continuity between the phases, documentation can be substantially reduced	Good for large and mission- critical projects	Each phase has specific deliverables
3.	Phases are processed and completed one at a time	Implementation of easy areas does not need to wait for the hard ones	Software is produced early in the software life cycle	Higher chance of success over the waterfall model due to the development of test plans earlier on during the life cycle
4.	Works well for smaller projects where requirements are very well understood	Works well for smaller and moderate size projects	Works well for projects where risk analysis contains higher priority	Works well for small projects where requirements are easily understood

Table 2: Comparing the disadvantages of different models [3].

S/N	Waterfall	Iterative Model	Spiral Model	V-Shaped Model
1.	Adjusting scope during the life cycle can kill a project	Milestones are more ambiguous than the waterfall	Can be a costly model to use	Very rigid, like the waterfall model
2.	No working software is produced until late during the life cycle	Activities performed in parallel are subject to miscommunication and mistake assumptions	Risk analysis requires highly specific expertise	Little flexibility and adjusting scope is difficult and expensive
3.	High amounts of risk and uncertainty	Unforeseen interdependencies can create problems	Project's success is highly dependent on the risk analysis phase	Software is developed during the implementation phase, so no early prototypes of the software are produced
4.	Poor model for complex and object-oriented projects. Poor model where requirements are at a moderate to high risk of changing	Changes are possible as it is an iterative model	Does not work well for smaller projects	Model does not provide a clear path for problem found during testing phases

2 | REVIEW OF LITERATURE

According to H. B. Mahapatra and B. Goswami [13], the researchers opined that research was carried out on software development methods like the iterative, prototype, RAD, Waterfall, XP and spiral using the various criteria; analysis of requirement, development team, the participation of the user, type of project and risk associated.

2.1 | REQUIREMENT ANALYSIS

An analysis was carried out on the following models; waterfall, prototype, iterative, spiral, RAD, and extreme programming using some criteria. Result obtained proofed that the waterfall model, the requirements are well defined and easily understood, it does not change often, at the starting iteration requirements are defined but does not indicate a multifarious system to be constructed. In the case of the prototype model, the requirements are difficult to know not very clear from the starting point, but requirements are changed often, it indicates multifarious system construction. In the case of the iterative model, requirements are not defined nor easily understood, do not changed often. However, requirements are defined at the starting of the iterations. The requirements show multifarious systems to be created. In the case of the spiral model, it is difficult to know and state the requirements even at the starting point, but requirements are uttered often, and it indicates a multifarious system to be created. For the RAD model, the requirement is well stated and easily understood, it is not uttered often, it is defined at the starting iteration, but it does not indicate a multifarious system to be created. Lastly, XP model requirements are not easily understood and defined even at the starting point, but it is changed often. It indicates a multifarious system to be created as shown in Table 3 [13].

2.2 | DEVELOPMENT TEAM STATUS

In the development team table, the status of the software development models namely; waterfall, prototype, iterative, spiral, RAD and XP is that development team requires less experience on similar projects for prototype and spiral model but for the waterfall, iterative, RAD and XP models do not require less experience. Also, less domain knowledge is required for waterfall, iterative, and spiral models but the prototype, Rad, and XP models do not require less knowledge. Slight experience on the tool is required for waterfall and spiral model while prototype, iterative, RAD, and XP do not require less experience. Availability of training is required for iterative, RAD, and XP model but is not required for the waterfall, prototype, and spiral models as shown in Table 4 [13].

2.3 | USER'S PARTICIPATION

In the user's participation, the different software development models like waterfall, prototype, iterative, spiral, RAD, and XP are such that the user's participation for all phases is required for prototype, RAD and XP models but not for the waterfall, iterative or spiral models. Limited user participation is required for waterfall, iterative, and spiral models but not required for prototype, Rad, and XP models. Users do not have previous experience of participation on a similar project for prototype, iterative, and spiral models but waterfall, Rad, and XP models, users have previous experience on similar projects. Users are experts of the problem domain for prototype, iterative, Rad, and XP models but are not experts for waterfall and spiral models respectively as shown in Table 5 [13].

Table 3: Analysis of requirement [13].

Analysis of Requirements	Waterfall	Prototype	Iterative	Spiral	RAD	XP
Understandable and definition of requirements are easy	Yes	No	No	No	Yes	No
Requirements are changed quite often	No	Yes	No	Yes	No	Yes
Requirements definition is at the starting of iterations	Yes	No	Yes	No	Yes	No
Multifarious system to be created is indicated by requirements	No	Yes	Yes	Yes	No	Yes

Table 4: Based on status of development team [13].

Development Team	Waterfall	Prototype	Iterative	Spiral	RAD	XP
Little experience on similar projects	No	Yes	No	Yes	No	No
Little domain knowledge (new to technology)	Yes	No	Yes	Yes	No	No
Little experience on tools	Yes	No	No	Yes	No	No
Training availability when needed	No	No	Yes	No	Yes	Yes

Table 5: Based on user's participation [13].

User's Participation	Waterfall	Prototype	Iterative	Spiral	RAD	XP
User participation in all phases	No	Yes	No	No	Yes	Yes
Limited User participation	Yes	No	Yes	Yes	No	No
User has no previous experience of participation in similar projects	No	Yes	Yes	Yes	No	No
Users are experts of the problem domain	No	Yes	Yes	No	Yes	Yes

2.4 | PROJECT TYPE AND ASSOCIATED RISK

In the type of project and risk associated, in terms of "project is the improvement of the old system" iterative, Rad, and XP models yielded "yes" while waterfall, prototype, and spiral produce "no". Funding is suitable for the project in the waterfall, prototype, and Rad but not suitable for iterative, spiral, and XP models. There are high-reliability requirements for iterative, spiral, and XP models but not for the waterfall, prototype, and Rad. The use of reusable components is applicable for prototype, spiral, RAD, and XP models but not for the waterfall, iterative, and XP models. Time, money and people are scares for prototype, and spiral models but not for waterfall, iterative, RAD nor XP models as shown in Table 6 [13].

Table 6: Type of project and risk associated [13].

Type of Project and Risk Associated	Waterfall	Prototype	Iterative	Spiral	RAD	XP
Project is the improvement of the old system	No	No	Yes	No	Yes	Yes
Stable funding for the project	Yes	Yes	No	No	Yes	No
Requirements are highly reliable	No	No	Yes	Yes	No	Yes
Schedule of the project is tight	No	Yes	Yes	Yes	Yes	No
Reusable components can be used	No	Yes	No	Yes	Yes	No
Scare resources (time, money, people, etc.)	No	Yes	No	Yes	No	No

Analysing Tables 3, 4, 5, and 6, it could be deduced that the characteristics associated with a particular type of method determine its appropriateness for a specific kind of project with no distinct user requirement, prototype, spiral, or XP methods are most suitable. A project with great risk involved, the spiral method is best suitable. It can control issues with the technicality of software applications subjected to different types of risks. The RAD model is most suitable for understandable project requirements with a specific schedule. An uncertain project with low risk and few team members the XP model is most preferable. Because many software developers encounter the problem of choosing the best suitable methodology for software projects. Selecting a model, it is determined by the different characteristics, no model is most suitable or right for a project. However, the researcher tried to compare some models considering the project characteristics. Again, it was stated that if there is any disparity during methodology selection the characteristics of the project must be ordered based on organizational culture and project situation. Furthermore, Y. S. Pundhir and B. Mishra [11] opined that the selection of a suitable model was based on the following: requirements, development team, users, project type, and associated risks.

2.5 | CHARACTERISTICS OF REQUIREMENTS

Requirements are very essential when choosing a suitable model. Some numbers of situations and problems during requirements capturing and analysis were stated in Table 7.

2.6 | DEVELOPMENT TEAM

In the case of the development team, availability, effectiveness, knowledge, intelligence, teamwork, etc., were seen to be very essential for project success. If the characteristics of the team are known, it will be easy to select a suitable model for the project. Table 11 shows development team status for the waterfall, V & V, spiral and incremental models [11].

Table 7: Characteristics of requirements [11].

Requirements	Waterfall	V & V	Spiral	Incremental
Easy to understandable and defined requirements	Yes	No	No	No
Requirements are changed quite often	No	Yes	Yes	Yes
Requirements are defined early in the cycle	Yes	Yes	No	Yes
A complex system created is indicated by requirements	No	Yes	Yes	Yes

Table 8: Status of development team [11].

Development Team	Waterfall	V & V	Spiral	Incremental
Similar projects with little experience	No	Yes	Yes	No
Knowledge in the domain is little	Yes	No	Yes	No
Tools to be used with little experience	Yes	Yes	Yes	No
Training if required is available	No	Yes	No	Yes

2.7 | PARTICIPATION OF USER

According to Y. S. Pundhir and B. Mishra [11], the involvement of the user increases their knowledge about the project. Therefore, user participation is an important role in choosing a suitable model. User's involvement is not required for the waterfall, V & V, spiral and incremental methods. Limited user participation is applicable in the four different models. No initial skill of users' participation in similar projects for spiral and incremental models but there is for waterfall and V & V models. Users are experts of the problem domain for V & V and incremental models but not the waterfall nor spiral model shown in Table 9.

Table 9: Participation of users [11].

Involvement of Users	Waterfall	V & V	Spiral	Incremental
In all phases, there are the participation of users	No	No	No	No
Participation of user is little	Yes	Yes	Yes	Yes
Users participation in similar projects without any skill	No	No	Yes	Yes
Experts of domain problem are the users	No	Yes	No	Yes

2.8 | TYPE OF PROJECT AND RISKS ASSOCIATED

Some models incorporate risk assessment and type of project as essential criteria in selecting a method for software development as shown in Table 10. In the end, the researcher posited that many methods are available for software developments despite the size. The waterfall method and the spiral method were utilized for developing systems. All methods have their benefits and demerits in developing software. Each method created was meant to improve on the previous method that was introduced. Presently, software organization chooses to implement the V-shaped model on their projects because of arrangement it follows to provide a bug-free product to client [11].

Table 10: Type of project and risks associated [11].

Type of Project and Risk Associated	Waterfall	V & V	Spiral	Incremental
An existing system enhancement is a project	No	No	No	Yes
Project funding is stable	Yes	No	No	No
Requirements are greatly reliable	No	Yes	Yes	Yes
Schedule of the project is tight	No	Yes	Yes	Yes
Use of reusable components	No	Yes	Yes	No
Scarcity of time, money, people resources	No	Yes	Yes	No

2.9 | RELATED WORKS

In this section, some research works which are related to the study were reviewed. It was discovered that different researchers have a different line of thought on the criteria to choose the best SDLC method for the success of a software project. V. Öztürk [14] opined that choosing the right SDLC could stimulate project growth. In terms of methodology, there is a tendency to minimize time, cost, minimize excesses, and any trace of risk, uncertainty management, quality assurance, user collaboration, and avail good project audit trail. Despite that, the ideal SDLC was used, selecting the right option was not easy. Some criteria could be considered for selecting suitable SDLC for project success namely; knowing requirements and changes are done, time of development, size of project, skill, risks, and complexity. An example was given that Iterative and IDD "incremental development" was preferable for a situation where the system intricacy was high. The researcher utilized a Fuzzy Logic (FL) system based on the criteria and the impact of choosing appropriate SDLCs in the different scenarios as shown in Table 11 [14].

Again, an FL system was developed to assist managers of project and developers of software application to choosing suitable SDLC(s). One billion (1.000) scenarios were used for selection at the same point of FL which was very hard and unsurmountable. The FL system was verified for different inputs shown in Table 12, the results obtained were; requirements are unstable and not well-known for agile, spiral, IDD, prototyping, and IDD-spiral models while the requirement was stable and well known for the waterfall model. Development time was little for agile, IDD and prototyping models, medium for spiral and IDD-spiral models, and lengthy for the waterfall model. Project size was seen to be small for both agile and prototyping models, medium for the spiral model, while large for IDD, IDD-spiral, and waterfall models. Expertise is experienced for agile, spiral, prototyping, and IDD-spiral models, inexperienced for the IDD model, and adequate for the waterfall model. The difficulty is high for spiral, IDD, prototyping, IDD-spiral, and waterfall models while medium for the agile model. The risk factor is large for agile, spiral, IDD, prototyping, and IDD-spiral models while medium for the waterfall model [14].

Table 11: Comparison of SDLC models [14].

Type of Project and Risk Associated	Waterfall	V & V	Spiral	Incremental
An existing system enhancement is a project	No	No	No	Yes
Project funding is stable	Yes	No	No	No
Requirements are greatly reliable	No	Yes	Yes	Yes
Schedule of the project is tight	No	Yes	Yes	Yes
Use of reusable components	No	Yes	Yes	No
Scarcity of time, money, people resources	No	Yes	Yes	No

Table 12: Fuzzy inference process outputs for different input values [14].

		Criteria						
No of FL Example	Input	Requirements	Development Time	Project Size	Expertise	Complexity	Risk	Output
1.Fuzzy inference	Input FL value Input FL number	Unstable and not well known 0.03/1	Short 0.05/1	Small 0.05/1	Experienced 0.9/1	Medium 0.5/1	High 0.95/1	Agile
2.Fuzzy inference	Input FL value Input FL number	Unstable and not well known 0.05/1	Medium 0.57/1	Medium 0.53/1	Experienced 0.978/1	High 0.97/1	High 0.97/1	Spiral
3.Fuzzy inference	Input FL value Input FL number	Unstable and not well known 0.02/1	Short 0.03/1	Large 0.98/1	Inexperienced 0.04/1	High 0.97/1	High 0.97/1	IDD
4.Fuzzy inference	Input FL value Input FL number	Unstable and not well known 0.02/1	Short 0.03/1	Small 0.056/1	Experienced 0.97/1	High 0.97/1	High 0.97/1	Prototyping
5.Fuzzy inference	Input FL value Input FL number	Unstable and not well known 0.05/1	Medium 0.57/1	Large 0.978/1	Experienced 0.978/1	High 0.97/1	High 0.97/1	IDD-Spiral
6.Fuzzy inference	Input FL value	Stable and well known	Long	Large	Adequate	High	Medium	Waterfall
	Input FL number	**0.9/1**	**0.9/1**	**0.95/1**	**0.5/1**	**0.9/1**	**0.5/1**	

According to A. Farrell [1], choosing a new growing method there is an effect in the company thereby requiring tactical and strategic planning to be considered. It is useful to include all stakeholders like customers outside the organization, executives, management, project teams, and shareholders in taking the decision and employees to ensure proper communication procedures in the organization. The size of an organization, project size, the number of projects, available resources, project team, their experience, and the customers determines which methodology is most ideal.

According to A. Farrell [1], most companies fall under an entrepreneurial structure. The recommended models for a commercial body are spiral, prototype, RAD, agile, and extreme programming. The iterative methods are most useful in situations with lots of alterations unlike formal methods, MDA, waterfall, ISO, CMMI, Adhoc and six-sigma methods cannot work in situations with large alterations shown in Table 13.

Table 13: Organizational structures and methodologies [1].

Model	Entrepreneurial	Innovative	Machine	Diversified	Professional
Waterfall	No	No	Yes	Yes	Yes
Spiral	Yes	Yes	Yes	Yes	Yes
Prototype	Yes	Yes	No	No	Yes
RAD	Yes	Yes	No	No	Yes
Agile	Yes	Yes	No	No	No
XP	Yes	Yes	No	No	No
Formal Methods	No	No	Yes	No	No
CMM/CMMI	No	No	Yes	No	Yes
ISO	No	No	Yes	No	Yes
Six Sigma	No	No	Yes	No	Yes
Ad-Hoc	No	No	No	No	No

According to J. Vermaet al. [15], it was opined that the selection of the exact software development method was a problem since the project has been rectified in another manner by some authors without any standard blueprint introduced. Linkert scale selection for the appropriate method was devised. A proficient system model for choosing the right method was given in Figure 2.

According to C. V. Geambaşu et al. [10], vital groups of development methodologies were reviewed with three, 3 methods namely RUP which is a "rational unified process", RAD, and XP. Some factors affect the choice of a method, the right methods to be adopted are affected by; initial requirements clarification, cost at the first stage, time of development, requirements incorporation, modifications integration at development stages, getting operational versions at system development, sensitiveness of software, cost of developing software, duration of delivery, difficult nature of the system, clients' and developers interaction, development team size. Legends used are low/small, medium, and high/large as shown in Table 14.

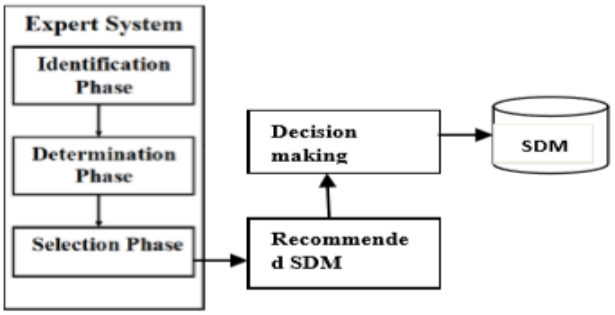

Figure 2: An expert system model [15].

Table 14: Factors that affect the choice of method for software development [10].

Factor	RUP	RAD	XP
F1: Initial requirements clarification	↑	→	↓ →
F2: Costs and development time initial accurate estimation	↑	→	↓
F3: Requirements changes integration at development stages	→	↑	↑
F4: During the development system functional version is obtained	→	↑	↑
F5: Sensitivity of software	↑	→	→
F6: Cost of development	↑	→	↓
F7: Final system length of delivery time	↑	↓	↓
F8: Complexity of the system	↑	↓ →	↓ →
F9: Customers verse developers' interaction	↑	→ ↑	↑
F10: Magnitude of the development team	↑	↓ →	↓

In Table 14, some conditions due to the assessment factor, a model was chosen as the best appropriate software development method. In another condition assessment of part of the factor resulted that a particular method was appropriate. A combination of two compatible models was the answer to the creation of software. In another scenario, collecting some parts of well-matched methods and putting them together to create software was the solution. In any of the scenarios choosing the right method is vital for the success of a project within a given time and budget [10].

Again in D. Thakur [16], the blueprint of software procedures is specific to the project. It is advisable to choose the method according to the software to be created. A project on software will be efficient if the method chosen was based on the requirement specifications. Also, period and price are vital and are very essential in choosing a method to be considered for a software project. In Table 15, choosing a method the elementary characteristics are the project type and associated risks. The vital feature of choosing a method is to know project reusability of its components, schedule, non-availability of resources, and risk associated [16].

Table 15: Selections on type of project and risks associated [16].

Project Type and Associated Risks	Waterfall	Prototype	Spiral	RAD	Formal Methods
Are requirements reliable?	No	No	Yes	No	Yes
Are the funds stable?	Yes	Yes	No	Yes	Yes
Are components reusable?	No	Yes	Yes	Yes	Yes
Is the schedule of the project tight?	No	Yes	Yes	Yes	No
Are resources scarce?	No	Yes	Yes	No	No

In Table 16, the vital feature of any method is to know the requirements of the project. In a situation with no clear definition of requirements by the user or lack of understanding of the developer the software created might not be efficient. In Table 17, the researcher opined that users needed to be consulted because the software was created for their use. When users are involved the clarity of the project will increase. It is most likely that the user is knowledgeable and has an idea of the requirements. Also, it might be that the user prefers a situation whereby the project is created sequentially such that it could be easily accessed to know the effectiveness at any point in time [16].

Table 16: Selection project requirements [16].

Project Requirements	Waterfall	Prototype	Spiral	RAD	Formal Methods
Early defined requirements in SDLC	Yes	No	No	Yes	No
Simply defined and clear requirements	Yes	No	No	Yes	Yes
Regularly requirements changes	No	Yes	Yes	No	Yes
Complex System due to requirements	No	Yes	Yes	No	No

Table 17: Selection on involvement of users [16].

Involvement of Users	Waterfall	Prototype	Spiral	RAD	Formal Methods
Little of user's involvement required	Y	N	Y	N	Y
User participation in all phases	N	Y	N	Y	N
No user experience of participating in similar projects	N	Y	Y	N	Y

According to R. L. Glass [17], a project where the management urged the team to achieve dates no matter what the management kept on adding more people to the team. The project was the least successful one but the project that the result functioned the way it was expected, developing the project was challenging and the team memes were small in size with good and high skill. The project was a success. The researcher stated that a new theory of software project success may be necessary because it will include real outcomes, taking into consideration the essence of a quality end product and organizational congruency. He stated that what constitutes success are the team members and management. R. Berntsson-Svensson and A. Aurum [18] posited that adding more personnel to meet schedule given is seen to contribute to the failure of a project while a well-defined scope of the project, complete and accurate necessities for the project, better estimations, the involvement of clients/users determines the success of a project. S. Abeet al. [19] opined that the factors that determine the success of a project are the quality of the result, the rate of development, and the period of the project. R. S. Dannelly and L. DeNoia [20] stated that a software development project is generally considered a success if it delivers the result at the expected period, within the stated budget, and comprises of all stated features or services. Y. Wang and F. Li [21] opined that the project manager's personality determines the success of a software development project.

J. Brown and J. T. Chao [22] opined that to ascertain the success of a project and know how it could be improved. Some assessment criteria were given such as quality of documentation. If it was prepared by the technical writer or the software development students to know which is better. If enough usability problems were resolved at the early stage of development. If the technical writer was able to note down the usability issues so that it's not overlooked until a later period when the problem will be more critical to handle and lastly the students' feedback. M. Pauline et al. [23] posited that performance measurement is used to manage and organize the quality of a project. It is a size gauge that is used during the development of projects on software to state, comprehend, gather, examine the data, and arrange it by checking and making some adjustments to better the product. One of the gauges is energy Effort Approximation used to manage the entire project costing and arrangement. Magnitude and density of the software are obtained from the function point analysis technique. In the research, fuzzy logic was used for measuring the value of necessities for the project, and the quality aspect is supplementary as a modification factor.

K. Kumar and S. Kumar [24] opined that choosing the appropriate SDLC model for a software project is directly proportional to the characteristics of the piece of software to be developed. All software is identified according to the characteristics with this the researcher introduced a rule-based recommendation system (RBR) that can aid in choosing the right SDLC model. A prototype developed was implemented for RBR. An ontology implementation of the taxonomy was stated for the proposed structural design of the RBR to use. It was given that in the future it is advisable to look at more software characteristics to propose a sustainable system. E. Mkoba and C. Marnewick [25] stated that many factors contribute to IT projects failing, such as a poorly defined project scope, insufficient human resources, cost overruns, poor communication among project stakeholders, project auditing practices not being followed, inadequate practices in project risk management and lack of correct auditing of processes in IT projects.

U. S. Shah [26] posited that the best way to design and sustain software was given by software engineers. SDLC is used to develop software projects by industries. It helped to declare necessities for the project, workable components, minimize design, maintenance cost, and avail robust software. There are three (3) categories of models namely traditional, agile, and hybrid models. The researcher focused on SDLC models, how to select the most effective SDLC considering some factors like kind of requirements, development team size, size of the project, and client's collaboration which influences the choice of criteria. A comparative analysis of different categories of models was performed and the analysis assisted the management to choose a suitable model for the software project. It was deduced that if the requirements for the project do not change if there are no devoted expert team members if the documentation of the project is very huge and the time expected to complete the project is much then the traditional waterfall model is best for the project. Also, it was incomprehensible to give the advantages of agile to traditional or hybrid to agile models due to the fact they these models are best used in some situations. Again, the research proved that the use and success of models depend on the mode of necessities of the project, the team, and their skills, magnitude, rate of amount, time, the conveyance period of the project, etc. Furthermore, the research assisted in selecting the best model with optimal performance in terms of the low amount involved and the period of completion of the entire work. The issues with these categories of models are lack of experts' full understanding of the domain area, poor collaboration among users, and improper use of the necessities for the project. Most developers prefer the hybrid model because it is a combination of the advantages of traditional

and agile models however the use of any models depends on the location or situation of the software.

According to M. Redka [27], the researcher posited that seven factors determine the bearing of a project irrespective of the methodology that will be finally adopted by the team of a company. These factors are namely requirements of the project, expected result, feedback from the team, rate of changes or improvements, delay price, team experience on method, and size of the project. L. Peters and A. M. Moreno [28] opined that experts have been able to recognize that a competent manager of a project is the most vital factor that determines the success of a software project. P. Mohagheghi and M. Jørgensen [29] posited that a successful project is a project that is seen to give the anticipated client profits. Again, all projects with adjustable scope and normal delivery were seen to be successful. Also, the participation of stakeholders with great importance and proper discussion between the client and provider contributes to the success of a project. More so, projects with period and material contracts and involvement of clients at the execution stage were seen to be successful. Success factors depend more on human factors, e.g., involvement, competence, and collaboration. Issues from human factors and technical nature must be resolved to have a successful software project.

According to R. Octavianus and P. Mursanto [30], the factors that determine the success of software development are scope, time, cost, performances, techniques and process, and finally client acceptance. Also, it was discovered that staff with adequate skills is the most inducing success factor in the development software. M. S. Jahan et al. [31] posited that the research analysis management of software projects in Pakistan. It was discovered that 90% of the information technology projects are contracted out. As a result, most foreign companies' residents in the country have a branch in Pakistan due to low employment opportunities. A good number of the project are from developed countries which required Software Project Manager (SPM) to handle the projects. As a result, there is a great need for SPM, and these brought about the increase in schools awarding certificate courses for SPM. An organization in all the SDLC phases may have an expert but if there is no skilled staff to manage it and nobody to discuss the goal of the company to obtain the set objectives at the end like the SPM who have the required skill of management of a software project then the project can fail.

3 | DISCUSSION ON DIFFERENT RESEARCH WORK

Discussions of the research work were generated from the findings obtained from the literature review chapter. According to H. B. Mahapatra and B. Goswami [13], the researchers used the option Yes or No to make some comparison. The researchers used the following models namely; waterfall, prototype, iterative, spiral, RAD and extreme programming to state that the decisions in selecting a method for software project were to use the following criteria namely; analysis of requirement, a team in the development, participation of users and type of project and risk associated. Another researcher, V. Öztürk [14] made a comparison of some SDLC models such as waterfall, iterative, and IID "incremental development", prototype, spiral, and agile. The criteria used were Known and Change of Requirement, Development Time, Project Size, Experience, Risk, and complexity. The legend for the comparison was as follows: H = High was blue, M = Medium was green, and L = Low was red.

Again, A. Farrell [1] made a comparison of the following system development model such as prototype, CMM/CMMI, waterfall, RAD, Ad-Hoc, spiral, XP, Agile, ISO, six-sigma, and formal methods using the following criteria entrepreneurial, innovative, machine, diversified, and professional. The scale used for the comparison was yes or no option. Also, Y. S. Pundhir and B. Mishra [11] made a comparison table with the following system development models such as V & V, spiral, incremental, and waterfall model. The researchers used some criteria which were the same as the criteria used by H. B. Mahapatra and B. Goswami [13]. The criteria are as follows: characteristics of requirements, the team in the development, involvement of users, type of project, and risks associated. The option used for the comparison was Yes or No.

Furthermore, C. V. Geambaşu et al. [10] made a comparison table using the following system development models namely: RUP, RAD, & XP with the following factors namely; Initial requirements clarification, costs and development time initial accurate estimation, requirements changes integration at development stages, during the development system functional version is obtained, the sensitivity of software, cost of development, final system length of delivery time, the complexity of the system, customers verse developers' interaction, and magnitude of the development team. The legends used are low, medium, high/large. Lastly, D. Thakur[16] made a comparison table using the following system development models namely: waterfall, prototype, spiral, RAD, and formal methods. The criteria the researcher used was the same as the criteria used by both works done by Y. S. Pundhir and B. Mishra [11] and H. B. Mahapatra and B. Goswami [13]. The criteria are as follows; selection on the type of project and risk associated, the selection on project requirements, and selection based on the user. Except selection based on the development team that was not treated.

3.1 | DIFFERENT AUTHORS WITH SIMILAR CRITERIA USED

During the study, the researcher discovered that different authors namely; H. B. Mahapatra and B. Goswami [13], Y. S. Pundhir and B. Mishra [11], and D. Thakur [16] came up with the same set of criteria for choosing the right SDLC for the success of software project as shown in Table 18.

4 | FINDINGS AND ANALYSIS

According to the findings from the tables of the following researchers, A. Farrell [1], Y. S. Pundhir and B. Mishra [11], H. B. Mahapatra and B. Goswami [13], and D. Thakur [16]. Detail comparison of the four different research work was organized in a single table comparison as shown in Table 19.

From Table 19, the "Y" represents yes while "N" represents no. Different criteria were used to check against different models. The "yes" response obtained indicated that for the particular criteria, the model was suitable for software projects while the "no" response indicated that for the particular criteria the model was not suitable for the software project. It was discovered that from the results obtained in Table 19, waterfall model yields nine (9) number of "yes" and fourteen (14) number of "no", prototype model yields thirteen (13) number of "yes" and ten (10) number of "no", iterative yields ten (10) number of "yes" and eight (8) number of "no", spiral yields sixteen (16) number of "yes" and seven (7) number of "no", RAD model yields twelve (12) number of "yes" and eleven (11) number of "no", XP model yields eight (8) number of "yes" and fifteen (15) number of "no", v & v model yields twelve (12) number of "yes" and six (6) number of "no", incremental model yields ten (10) number of "yes" and eight (8) number of "no, formal method yields eight (8) number of "yes" and nine (9) number of "no" and agile model yields two (2) number of "yes" and three (3) number of "no". Therefore, since the highest amount of yes obtained was spiral which gave sixteen (16) number of "yes". This indicated that the spiral model was the most suitable method for a software project and the least was the agile model with two (2) numbers of "yes" responses.

Table 18: Different researchers with similar criteria used.

H. B. Mahapatra and B. Goswami [13]	Y. S. Pundhir and B. Mishra [11]	D. Thakur [16]	A. Farrell [1]
Analysis of requirement	Requirement	Project Requirements	-
Understandable and definition of requirements are easy	Easy to understandable and defined requirements	Simply defined and clear requirements	-
Requirements are changed quite often	Requirements are changed quite often	Regularly requirements changes	-
Requirements definition is at the starting of iterations	Requirements are defined early in the cycle	Early defined requirements in SDLC	-
Multifarious system to be created is indicated by requirements	The complex system created is indicated by requirements	Complex System due to requirements	-
Development Team	Development Team	-	-
Little experience on similar projects	Similar projects with little experience	-	-
Little domain knowledge (new to technology)	Knowledge in the domain is little	-	-
Little experience on tools	Tools to be used with little experience	-	-
Training availability when needed	Training if required is available	-	-
User's Participation	Involvement of Users	Involvement of Users	-
User participation in all phases	In all phases, there are the participation of users	User participation in all phases	-
Limited User participation	Participation of user is little	Little of user's involvement required	-
User has no previous experience of participation in similar projects	Users participation in similar projects without any skill	No experience of participating in similar projects	-
Users are experts of the problem domain	Experts of domain problem are the users	-	-
Type of Project and Risk Associated	Type of Project and Risk Associated	Project Type and Associated Risks	-
Project is the improvement of the old system	An existing system enhancement is a project	-	-
Suitable funding for the project	Project funding is stable	Are the funds stable?	-
Requirements are highly reliable	Requirements are greatly reliable	Are requirements reliable?	-
Schedules of the project are tight	Schedule of the project is tight	Is the schedule of the project tight?	-
Reusable components can be used	Use of reusable components	Are components reusable?	-
Scare resources (time, money, people, etc.)	Scarcity of time, money, people resources	Are resources scarce?	-
-	-	-	Organizational Structures and Methodology
-	-	-	Entrepreneurial
-	-	-	Innovative
-	-	-	Machine
-	-	-	Diversified
-	-	-	Professional

Table 19: Combined table results from A. Farrell [1], Y. S. Pundhir and B. Mishra [11], H. B. Mahapatra and B. Goswami [13], and D. Thakur [16].

	Software Development Models									
Criteria	**Waterfall**	**Prototype**	**Iterative**	**Spiral**	**RAD**	**XP**	**V & V**	**Incremental**	**Formal Method**	**Agile**
Simply defined and clear requirements	Y	N	N	N	Y	N	N	N	Y	
Regularly requirements change	N	Y	N	Y	N	Y	Y	Y	Y	
Early defined requirements in SDLC	Y	N	Y	N	Y	N	Y	Y	N	
Complex System due to requirements	N	Y	Y	Y	N	N	Y	Y	N	
User's experience on similar projects	N	Y	N	Y	N	N	Y	N	Y	
Less domain knowledge (new technology)	Y	N	Y	Y	N	N	N	N		
Less experience on tools to be used	Y	N	N	Y	N	N	Y	N		
Availability of training requirement	N	N	Y	N	Y	Y	Y	Y		
Users participation in all phases	N	Y	N	N	Y	Y	N	N	N	
Participation of user is little	Y	N	Y	Y	N	N	Y	Y	Y	
Users participation in similar projects without any skill	N	Y	Y	Y	N	N	N	Y		
Experts of domain problem are the users	N	Y	Y	N	Y	Y	Y	Y		
Project is the improvement of the old system	N	N	Y	N	Y	Y	N	Y		
Are the funds stable?	Y	Y	N	N	Y	N	N	N	Y	
High-reliability requirement	N	N	Y	Y	N	Y	Y	Y	Y	
Tight project schedule	N	Y	Y	Y	Y	N	Y	Y	N	
Use of reversible components	N	Y	N	Y	Y	N	Y	N	Y	
Is resource (time, money, people, etc.) scarce?	N	Y	N	Y	N	N	Y	N	N	
Entrepreneurial	N	Y		Y	Y	Y			N	Y
Innovative	N	Y		Y	Y	Y			N	Y
Machine	Y	N		Y	N	N			Y	N
Diversified	Y	N		Y	N	N			N	N
Professional	Y	Y		Y	Y	N			N	N
Total Occurrence	**Y=9 N=14**	**Y= 13 N=10**	**Y= 10 N=8**	**Y= 16 N=7**	**Y= 12 N=11**	**Y=8 N=15**	**Y=12 N=6**	**Y=10 N=8**	**Y=8 N=9**	**Y=2 N=3**

5 | CONCLUSION

This research work presents the criteria for the right SDLC method for the success of software project as follows; "simply defined and clear requirements", "regularly requirements changes", "early defined requirements in SDLC", "complex system due to requirements", "user's experience on similar projects", "less domain knowledge (new technology)", "less experience on tools to be used", "availability of training requirement", "users participation in all phases", "participation of user is little", "users participation in similar projects without any skill", "experts of domain problem are the users", "project is the improvement of the old system", "are the funds stable?", "high-reliability requirement", "tight project schedule", "use of reversible components", "are resource (time, money, people etc.) scarce?", "entrepreneurial", "innovative", "machine", "diversified", and lastly "professional". Based on discoveries from the research as regards the criteria with different models presented. Spiral model was the most suitable method for the success of software projects with the result obtained as "yes" to be sixteen (16) and "no" to be seven (7), next was prototype model with thirteen (13) number of "yes" and ten (10) number of "no", followed by v & v model with twelve (12) number of "yes" and six (6) number of "no", next was RAD model with twelve (12) number of "yes" and eleven (11) number of "no", followed by the iterative or incremental model with ten (10) number of "yes" and eight (8) number of "no", next was waterfall model with nine (9) number of "yes" and fourteen (14) number of "no", followed by formal method with eight (8) number of "yes" and nine (9) number of "no", next was XP model with eight (8) number of "yes" and fifteen (15) number of "no" and the least was agile model with two (2) number of "yes" and three (3) number of "no". This implied that irrespective of the criteria stated for all the models, spiral yielded the maximum numbers of "yes" to be the most suitable method to adopt for any software project. The limitations were that out of so many models available only ten (10) were used with the criteria presented for the analysis. To obtain a more reliable and standard result, it is better to use more models against the given criteria to obtain an optimal result and deductions of the most suitable method to be adopted.

REFERENCES

[1] A. Farrell, "Selecting a software development methodology based on organizational characteristics," Athabasca University, 2007.

[2] S. K. Dora and P. Dubey, "Software Development Life Cycle (SDLC) Analytical comparison and survey on Traditional and agile methodology," *Abhinav National Monthly Refereed Journal of Research in Science & Technology,* 2013.

[3] V. Rastogi, "Software development life cycle models-comparison, consequences," *International Journal of Computer Science and Information Technologies,* vol. 6, no. 1, pp. 168-172, 2015.

[4] M. Sami. "Software Development Life Cycle Models and Methodologies." https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/ (accessed May 23, 2019).

[5] P. Frijns, R. Bierwolf, and T. Zijderhand, "Reframing security in contemporary software development life cycle," in *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)*, 2018: IEEE, pp. 230-236.

[6] Y.-H. Tung, S.-C. Lo, J.-F. Shih, and H.-F. Lin, "An integrated security testing framework for secure software development life cycle," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2016: IEEE, pp. 1-4.

[7] S. Kaur, "A review of software development life cycle models," *International journal of advanced research in computer science and software engineering,* vol. 5, no. 11, pp. 354-360, 2015.

[8] J. Sivaranjani and S. Rajeswari, "A Study on Software Development Methodologies," *International Journal of Innovative Research in Computer and Communication Engineering,* vol. 5, no. 4, pp. 7727-7737, 2017.

[9] E. M. Simão, "Comparison of software development methodologies based on the SWEBOK," 2011.

[10] C. V. Geambaşu, I. Jianu, I. Jianu, and A. Gavrilă, "Influence factors for the choice of a software development methodology," *Accounting and Management Information Systems,* vol. 10, no. 4, pp. 479-494, 2011.

[11] Y. S. Pundhir and B. Mishra, "Various SDLC Models & Their Different Usage Criteria are During the SDLC Model Selection Process at the Time of Software Projects Initiations and Development," *Bookman international journal of software engineering,* vol. 1, no. 1, pp. 19-23, 2012.

[12] X. Zhang, T. Hu, H. Dai, and X. Li, "Software development methodologies, trends, and implications," *Information Technology Journal,* vol. 9, no. 8, pp. 1747-1753, 2010.

[13] H. B. Mahapatra and B. Goswami, "Selection of software development methodology (SDM): a comparative approach," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 5, no. 3, pp. 58-61, 2015.

[14] V. Öztürk, "Selection of appropriate software development life cycle using fuzzy logic," *Journal of Intelligent & Fuzzy Systems,* vol. 25, no. 3, pp. 797-810, 2013.

[15] J. Verma, S. Bansal, and H. Pandey, "Develop framework for selecting best software development methodology," *International Journal of Scientific & Engineering Research,* vol. 5, no. 4, p. 1067, 2014.

[16] D. Thakur. "Criteria for Selecting Software Process Models." https://ecomputernotes.com/software-engineering/criteria-for-selecting-software-process-models (accessed May 23, 2019).

[17] R. L. Glass, "Evolving a new theory of project success," *Communications of the ACM,* vol. 42, no. 11, pp. 17-19, 1999.

[18] R. Berntsson-Svensson and A. Aurum, "Successful software project and products: An empirical investigation," in *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*, 2006, pp. 144-153.

[19] S. Abe, O. Mizuno, T. Kikuno, N. Kikuchi, and M. Hirayama, "Estimation of project success using Bayesian classifier," in *Proceedings of the 28th international conference on Software engineering*, 2006, pp. 600-603.

[20] R. S. Dannelly and L. DeNoia, "Student opinions of software project success," in *Proceedings of the 45th annual southeast regional conference*, 2007, pp. 327-330.

[21] Y. Wang and F. Li, "How does project managers' personality matter? building the linkage between project managers' personality and the success of software development projects," in *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, 2009, pp. 867-874.

[22] J. Brown and J. T. Chao, "Collaboration of two service-learning courses: Software development and technical communication," *Issues in Informing Science and Information Technology,* vol. 7, pp. 403-412, 2010.

[23] M. Pauline, P. Aruna, and B. Shadaksharappa, "Fuzzy-Based Approach Using Enhanced Function Point to Evaluate the Performance of Software Project," *IUP Journal of Computer Sciences,* vol. 6, no. 2, 2012.

[24] K. Kumar and S. Kumar, "A rule-based recommendation system for selection of software development life cycle models," *ACM SIGSOFT Software Engineering Notes,* vol. 38, no. 4, pp. 1-6, 2013.

[25] E. Mkoba and C. Marnewick, "IT project success: A conceptual framework for IT project auditing assurance," in *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, 2016, pp. 1-8.

[26] U. S. Shah, "An excursion to software development life cycle models: an old to ever-growing models," *ACM SIGSOFT Software Engineering Notes,* vol. 41, no. 1, pp. 1-6, 2016.

[27] M. Redka. "7 Things to Pay Attention to When Choosing a Software Development Methodology." https://mlsdev.com/blog/136-7-things-to-pay-attention-to-when-choosing-a-software-development-methodology (accessed July 10, 2019).

[28] L. Peters and A. M. Moreno, "Evaluating Software Project Managers: A Multidimensional Perspective," *IEEE Software,* vol. 34, no. 6, pp. 104-108, 2017.

[29] P. Mohagheghi and M. Jørgensen, "What contributes to the success of it projects? success factors, challenges and lessons learned from an empirical study of software projects in the norwegian public sector," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, 2017: IEEE, pp. 371-373.

[30] R. Octavianus and P. Mursanto, "The Analysis of Critical Success Factor Ranking for Software Development and Implementation Project Using AHP," in *2018 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 2018: IEEE, pp. 313-318.

[31] M. S. Jahan, M. T. Riaz, K. S. Arif, and M. Abbas, "Software project management and its tools in practice in IT industry of Pakistan," in *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 2019: IEEE, pp. 1-6.