https://dx.doi.org/10.4314/dujopas.v10i1b.27
ISSN (Print): 2476-8316
ISSN (Online): 2635-3490

Dutse Journal of Pure and Applied Sciences (DUJOPAS), *Vol. 10 No. 1b March 2024*

Software Engineering in Nigeria: Trends and Challenges

[1]Abdullahi Anas*, [2]Aminu Labaran, [3]Muhammad Abullahi Bakale, [1]Bashar Aliyu, [1]Halimatu Abubakar Ainu, [4]AbdulAzeez Abdullahi Bena, [1]Yusuf Umar Jimada

[1]Department of Computer Science,
Faculty of Sciences
Sokoto State University,
Sokoto – Nigeria.

[2]Department of Computer Science
Faculty of Sciences
Federal Polytechnic Kaura Namoda,
Zamfara state – Nigeria

[3]Department General Studies,
Collage of Agriculture and Animal Science,
Wurno Sokoto– Nigeria.

[4]Department of Computer Science
Faculty of Sciences
Waziri Umaru Federal Polytechnic
Birnin Kebbi – Nigeria

Email: anas.abdullahi2@ssu.edu.ng

Abstract
Bringing theory and practice together is aspiration of software engineering education, so that the learner may acquire a deep understanding on basic concepts and principles, along with skills and competences to solve real-world problems. Thus, it is not just an academic concern about teaching relevant topics, but also a task to shape skilled individuals equipped for the industry demand. Moreover, due to the rapid development the software industry is and transnational nature, several challenges are challenging the qualification of software engineers, capable to develop products according to the international standards into markets overseas. Agile software development methods are the generally used and prominent software development methods. Agile methods are simple, easy and deliver software faster than the traditional software development methods. Software development in Nigeria is at developing stage.

Keywords: *Software development, agile, waterfall, agile manifesto*.

INTRODUCTION
Software Engineering gives the procedures and practices to be followed in the software development (Abrahamsson *et al.,* 2021 & Anas *et al.,* 2023a). To computer science engineering techniques, software engineering acts as a backbone. Software engineering is a discipline that undergone many improvements that aims to keep up with the new advancements in

**Author for Correspondence*

technologies and the modern business requirements through developing effective approaches to reach the final software product (Alexander *et al.,* 2023 & Anas *et al.,* 2023b). Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. Software engineering provides theories, steps and procedures to be followed and applied in the process of software development. These constitute the software development methodologies. The software development approaches can be broadly classified into heavyweight and lightweight approaches (Braude *et al.,* 2019 & Anas *et al.,* 2023c). The heavyweight approaches which are also known as traditional approaches includes waterfall, iterative incremental and evolutionary approaches among others. The approaches are applied in development of huge and complex software. The developments in computer technology, creates and sets new requirements and goals in software development process. Adapting to these challenges, currents methods has to be improved and optimized. The rigid conventional software development methods were incapable of adapting to these changes which leads to the development of light weight software development methods. The lightweight approach which is also called agile methodology is a conceptual framework for software engineering that begins with a starting planning phase and follows the road toward the deployment phase with iterative and incremental interactions throughout the life-cycle of the project (Anas *et al.,* 2022). In 2020, an official association of 17 software engineering consultants resulted in the publication of Agile Software Manifesto (Jammalamadaka *et al.* 2021) in which a set of values and principles in software and system agility are outlined.

AGILE DEVELOPMENT MANIFESTO

The sets of values and principles in software and system agility outlined states that in any situation where a choice has to be made, a priority is given toward the items on the left of each core of its value rather than the ones in its right (Kashumi *et al.* 2023). These items are as follows;

i. Individuals and interactions over processes and tools
ii. This value put emphasis on the importance of communication, interaction and the quality of the human software developers that these factors serve over the abstract formal processes and their technical surrounding environment as a key factor in the software development.
iii. Working software over comprehensive documentation

As documentations are rarely used beyond deployment phase, the amount of time, consideration and resources to be allocated to software documentation should not overwhelm the software development process. Documentation should be optimized and controlled. Working software should be the highest priority in the software development process.

iv. Customer collaboration over contract negotiation

As a lightweight development process, agile software development processes are designed to adapt to the changes in the customer requirements during the development process. Frequent feedback, negotiation and collaboration between customer and the development team is given higher priority over contract negotiation as new customers' requirements are captured earlier at any stage.

v. Responding to change over following a plan

With software development process proceeding, both the two stakeholders (developer and customer) gains more knowledge and understand the system better, thus, requirements refinement may be necessary. The significance in the agile manifesto is given to the reaction to the change in the development process life cycle instead to follow a strict defined blueprint, because the ultimate goal to reach is the customer satisfaction.

AGILE DEVELOPMENT METHOD

Agile development method is an umbrella of different processes that supports the agile values and principles. Each method comprises of different combinations of procedures on how daily work is done by each development team member. Every single method has its peculiar notions, life cycle, functions, merits and demerits etc. These methods assemble the software in an iteratively incremental process.

Feature driven development (FDD) method

As one of the agile development methods, FDD handles short incremental iterations that lead to functional software. To the required software, feature is a valued function for the user. Jeff De Luca first introduced the idea of FDD in 1997. He used this idea in a big project after realizing that the other development methods are not convenient in developing large projects over a specific time. The FDD's main idea is to control the software development focusing on the requirement feature list in the business needs. FDD is an adaptively high software development method which accepts late changes of software requirements. The main emphasis of FDD is delivery of high quality outputs throughout the development process phases.

Feature driven development (FDD) life cycle:

Feature driven development life cycle contains five sequential processes (Anwer *et al.* 2017 & Abrahamsson *et al.* 2021), these processes are performed in an incremental iterative way where they will deliver the final software. They are:

a) Overall Model Development:

The overall required project framework and span is explicitly defined by all the team members and the experts in this stage. Numerous models can be derived from different teams and experts. These models are examined and the most optimal model for the project based on requirements is selected.

b) Feature List Building:

In this step the derived model and its requirement documentation are used to construct a list of system features that the users' wants in the system. This feature list will be revised by the customer as well as experts of the business and finally confirmed.

c) Plan by feature:

This step creates an advanced plan derived from the previously approved feature list. It will be structured based on the priority to the customer and inter-features dependency. It contains major milestones schedules for the project and a detailed schedule feature. Both managers (project and development), and the chief programmer are occupied in this step. The chief programmer allocates the features to a particular developer as class owners.

d) Design by feature:

As an iterative step, each iteration tracks few days but two weeks at most. The Chief programmer and class owners construct a design package for each class, along the sequence diagrams. These design packages and diagrams are go through for approval.

e) Build by feature:
In the FDD process this is the final step, coding, code inspection and testing processes are all carried out. Similar to the design by feature step, this step is also an iterative step, the developed features will be published in the main build, upon completion of the iterations then a new feature set is started and so on.

FDD roles:
There are six (6) key roles in FDD method, each can be carried out by more than one person, and one team member can do many roles. The roles include:

i. Provision of administrative decisions and the best working environment by the Project Manager.
ii. Overall design approval of the requested software design done by chief architect.
iii. Development team supervision by the development manager.
iv. Control, coordination and management of the development process.
v. Features design, coding, and testing by class owners.
vi. Provision of technical knowledge about the business which will enrich the requested features list to be implemented by the domain expert.

Table 1, Show the Advantages and Disadvantage of FDD roles

Advantages	Disadvantage
Accepts late changes by the customer	No guidance about the requirement gathering
Quick feedback from the customers	Critical issues of the projects are not addressed
High-quality result after each phase	

Test Driven Development (TDD) method
This method is based on building a small iteratively automated for testing programs, then to write the code that can pass that test, and leave the enhancement of that code to be done later. TDD is considered as the opposite of the traditional software development methods which do the test after the code is done. The idea of Test Driven Development was introduced in 2003 by Kent Beck, but it was already used by NASA in 1950 while they are working in the Mercury project. By following the Test Driven Development the fault and the bugs' rate will be decreased and will enhance the code quality. Test Driven Development has two main rules these are

i. If the test fails, then write a code to solve it.
ii. Don’t make duplications in the code.

The written tests by the developers will not be considered as the only needed test for the project, because of the developer write a single testing program that is needed in a small required functionality in the software. So, the software still needs to be tested using other types of tests such as the stress test, performance test etc.

Test Driven Development life cycle: The Test Driven Development life cycle contains five main steps as discussed, these set of steps will be done on each feature of the required software. After these steps are finished the resulted software will be high-quality software, these steps are:

i. Add a test; as we mentioned before, the first step in developing software using the TDD method is to prepare an automated test based on the requirement of the software. This automated test is written by the developer, he uses the customer requirements list and the use cases for these requirements.
ii. Run all test cases on the code which is not available on this step, so all the test cases will fail to achieve some requirements.

iii. Write the code, the developer will review the failed test cases and write the code that solves these test cases. The developer main objective in this stage is to pass the test, so the resulted code may not be efficient but it will be refined later.
iv. Run all the test cases again, the developer will rerun all the test cases on the developed code, to check if the problem is solved or not. If the code is passed on all test cases, then the new code is satisfying the requirements needed. If not, then the code needs to be modified again and again until it will be passed.
v. Refactor the code, this step is the last step in the TDD, in this step the code will be refined, and removed the duplications. The code may be changed; but these changes must not affect the main objective of it, this means that the code must still pass the tests and the modifications do not effect on its main functionality.

Table 2, Show the Advantages and Disadvantage of TDD method

Advantages	Disadvantage
Early detection of defects and bugs	Leads to poor documentation
High quality code	Repeated test failures leads to being struck
Easy development	Demands extra skills from programmers
Reduced complexity	

Dynamic System Development Method (DSDM)
Dynamic System Development Method or Model (DSDM), this method uses the rapid application development approach (RAD). It is an incremental iterative approach where the quality of the software is very critical value in this method. The idea of DSDM was in 1994 by practitioners of a consortium in the United Kingdom, then it became a framework for the rapid applications in 1997. The main idea about the DSDM is to set and define the resources of the project and the time for the project and then adjust the amount of functionality that can be done in these time and resources. It is the opposite of the traditional methods where the list of functionalities required in the system are predefined first then the time and the resources will be allocated and defined based on these resources. (Anwer *et al.* 2017 & Abrahamsson *et al.* 2021).

DSDM life cycle: Dynamic System Development Method contains five main phases. DSDM as we discussed before it is an incremental method; so, the feedback of clients is directed processed to improve the quality of the software. (Anwer *et al.* 2017 & Abrahamsson *et al.* 2021) outlined DSDM phases. These are:
i. Feasibility study phase which is the first step in the DSDM method, in this step the project will be studied to decide if the DSDM is appropriate to use in the development or not. Also, the risk analysis and specifying the technical requirements are done in this phase. The feasibility report and the outline plan for development are the results from this step.
ii. Business study phase, in this phase meetings are done between the business experts from the client side and the development team, these meetings are done to specify the user's requirements and to make a list of functionalities needed in the system with its priorities. Technology which will be used is specified in this step, also a high-level description of the process, ER-Diagrams, object mode, system architecture, system architecture and the outline prototyping plan are the results of this step.
iii. Functional model iteration, in this phase the analysis, coding and prototyping are done iteratively and incrementally, the prototypes are analyzed to enhance the analysis model. The main objective of this step is to specify what will be developed in the software, when it will be developed and the development way. Also, the prioritized functions, functional prototyping, nonfunctional requirements and the risk analysis.

iv. Design and build phase, this phase is also an iterative phase, in this phase, the identified requirements from the previous phase is implemented and coded, and then it will be published and tested by the users. The user's feedback is used to enhance the system iteratively. So, tested software with an at least minimal set of the requirement is the result of this step.
v. Implementation phase, in this phase the software moved from the development to the production and it will be handled to the users, training sessions are provided. Also, the user manual is developed and handled. If the software is fulfilling all the user's requirements, then there are no other development processes is needed. If not, the whole process is redone again.

DSDM roles:
There are many different roles used in dynamic software development method, the primary roles are as following:

i. The developers who are all the development team including the senior developer, analyst, designer, programmer and the tester.
ii. Technical coordinator who is the person who ensures that the business and the technical aspects of the project are following the plan. He defines the architecture of the system and ensures the technical quality of the project.
iii. Ambassador user who is from the customer side, he is the person who will use the developed system; he should be able to discuss all the users' needs to the development team.
iv. Advisor user who is also from the customer's side, he can provide some important viewpoints that is needed in the project where the ambassador user can't provide.
v. Visionary who is the user who has a good understanding of the whole business needs and objectives. He assures that the most important business requirements are provided to the developers and are done in the right way.
vi. Executive sponsor is the person from the customer side who has the power of making any decision and has the authority on the financial issues.

Table 3. Show the Advantages and Disadvantage of DSDM roles

Advantages	Disadvantage
Provides rapid application development	Difficulties in the administration of the project
Provide guidelines for the project aspects	Does not consider the project criticality

Scrum method
Schwaber & Beedle (2002) proposed a project management called scrum method. The method is aimed for iterative software development process. Scrum is a concrete implementation of an agile framework. Deliverance of the highest value in the shortest time is the main focus of the method. It is a team oriented agile methodology that specifies a certain role, establishes a short time boxed iteration called sprints in which the system is incrementally developed and produces a different artifact that coordinates its work.. Scrum is among the most used agile methods. Scrum focus on software management issues and its simplicity makes among the most popularly used method.

Table 4, Show the Advantages and Disadvantage of Scrum method

Advantages	Disadvantage
Transparency	No precisely defined responsibilities
Self-organization	Does not prescribe any specific practices
Self-retrospective	
Simple process	

Extreme programming method

Extreme programming (XP) this is one of the earliest proposed agile methods. To overcome the shortcomings of the convolutional software development process caused by the frequent requirements changes, suit object-oriented project comprising of multiple programmers on a single node (Beck & Gamma 2022) proposed the XP agile method. It is a mixture of prominent software engineering practices. XP reduces the cost of the requirements changes through long development cycle replacement with multiple short cycles to achieve a customer satisfaction. This takes the concepts of software engineering to an extreme level which in return improves software quality. Beck (2022) outline 13 primary XP technical practices which are: sit together, whole team, informative workspace, energized work, pair programming, stories, short iteration, quarterly release, slack, ten minute build, continuous integration, acceptance- and unit test-driven development, and incremental design, in an addition of 11 corollary practices: root cause analysis, collective code ownership, code and tests, negotiated scope contract, real customer involvement, incremental deployment, team continuity, shrinking team, daily deployment, single code base, pay-per-use practices. These practices leads to the development of five key values: communication, simplicity, feedback, courage, and quality work (Williams 2010).

XP life cycle: The life cycle of XP consists of six phases (Abrahamsson *et al.* 2021) that can be explained below:

1. The Exploration phase:
   i. The customer is an essential part in the XP team, he is responsible for making decisions about the requirements and features that are expressed as user stories that they think must exist in the first release.
   ii. The project team gets introduced to the available resources such as the tools, technology and practices in order to be familiar with them in the project.
   iii. A sample prototype of the system is built to test the technology and to discover the possible architecture of the system.
   iv. Duration: A few weeks to a few months.
2. The Planning phase:
   i. Effort estimation and the schedule are made and agreed upon by the programmers.
   ii. The stories are prioritized.
   iii. Duration: A couple of days.
3. The Iterations to Release phase:
   i. It includes several iterations with each iteration lasts from one up to four weeks.
   ii. The first iteration is a special iteration with the goal of creating an overall system architecture by selecting the appropriate stories.
   iii. The functional tests are developed by the customer and ran at the end of every iteration. The last iteration output will be ready for a production system.

Two main practices in this phase: pair programming and refactoring: Pair programming and Refactoring; Pair programming, in which the developers work in pairs to develop the code, i.e., one is actively writing the code, and the other observes, supports and reviews that code. This has many benefits such as spreading the knowledge across the team especially that these pairs are selected dynamically, and developing a collective ownership and common responsibility of the developed code. Moreover, it supports refactoring to improve the software. Refactoring is a tool to improve the software and makes it simple and maintainable by reconstructing its code without changing its functionality, for example by removing duplication, adding flexibility, or renaming the code variables and functions for a better understanding.

4. The Productizing phase:
   i. An additional testing and evaluating of the performance of the system are needed before it can be released to the customer.
   ii. After the first release of the system is delivered to the customer, the system should be kept running while new iterations are produced.
5. The Maintenance phase: incorporating new people into the team may be required to support customer tasks.
6. The Death phase: In this phase, no additional user stories existed, thus no more changes are requested and the current system implementation satisfy the user requirements and needs. Furthermore, Death may occur if the system stops delivering the desired results, or if any further development is costly.

XP roles: There are different roles in XP for different tasks:

i. **Programmer**: Is responsible for writing simple, high quality functioning codes, usually done by the collaboration with other programmers according to the pair programming practice.
ii. **Customer**: The customer writes the system requirements and features as stories and functional tests, and assign them the appropriate priorities and decides at the end of the process if each requirement is satisfied or not.
iii. **Tester**: Testers assist the customer in writing the functional required tests and run the tests.
iv. **Tracker**: Tracker analyze the team estimations and progress in each iteration and provide them with his feedback.
v. **Coach**: Is responsible for guiding the overall process.
vi. **Consultant**: Consultant is qualified outside the team member that helps the team in solving any encountered problems.
vii. **Manager**: A manager is capable of making decisions and responsible for the communication and elimination of any obstacle in the team.

SOFTWARE DEVELOPMENT CHALLENGES IN NIGERIA.

There is plethora of challenges faced by software development IT industry of Nigeria which includes incompetent developers, unfavorable market, bad government policies, and corruption among others.

1. Incompetent developers

The current state of the Nigerian education sector is totally not in favor of software development sector of the Nigerian IT industry. Inadequate laboratories, research grant, strikes and outdated curriculums. Majority of Nigerian graduates of the software development sector which includes computer science, software engineering, computer engineering and ICT lacks software development education which is a vital aspect of the software development process.

2. Bad government Policies

The Nigerian government enacts laws that totally affect the software development sector directly and indirectly. For example, with the unfavorable market in Nigeria, developers tends to sell their products in the global market where they usually got paid in dollar, Nigerian government enact laws that make receiving payment outside Nigeria very difficult through such platforms such as paypal. Also Nigerian government enact laws that limits international dollar spending through banks to $20 per month which makes it difficult to for developers to pay for online courses offered abroad.

3. **Corruption**

   One of the leading dark forces setting Nigeria behind and making sure it never progress is corruption. It is found in every sector and our homes. Like in other sectors, corruption had greatly impacted the software development sector of the Nigerian industry. Contracts are awarded to incompetent, greedy and unpatriotic bidders. Research grant allocated for the development of the sector and Nigeria are embezzled by the officials that are in controlled of the area.

CONCLUSION

The technological advancement of 21st century has led to the massive development of economies of developing countries and turns them into developed and super economies. This is realized as a result of their massive investment in technology. Countries like Japan, Malaysia, Singapore, India and China are now the front runners of the IT industry. With the world now geared towards Artificial Intelligent (AI), it is obvious now to say countries with best investment in technology will be the leading economy in the coming decades. Investment in technology is simply investing software engineering education. As an umbrella of prominent development processes, agile methodology is the widely used and acceptable software development process.

REFERENCES

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2021). Agile software development methods: Review and analysis, *arXiv preprint arXiv:1709.08439*.

Alexander M., Mohammed N & Moe H (2023). Comparing Measured Agile Software Development Metrics Using an Agile Model Based Software Engineering Approach Versus Scrum Only. *IEEE Transaction on software Engineering, 406-419.*

Anas A., JimadaY. U., Muhammed A. U., Aliyu A. A., Adamu B. I., & Zaharadeen Y. Y.(2023a). DRX Mechanism for Machine Type Communication (MTC) in Long Term Evolution Network (LTE): Potentials and challenges. *Dutse Journal of Pure and Applied Sciences* (DUJOPAS) 9 (4a): 227-235, *https://dx.doi.org/10.4314/dujopas.v9i4a.21*.

Anas A., Mansur A., Adamu B. l., Muhammad A. B., Zaharadeen Y. Y., Abdullahi S., Aliyu A., & Abdulmalik A. (2023b). Safe Campus: An Intelligent Campus Video Surveillance System for Crowd Analysis. *Dutse Journal of Pure and Applied Sciences* (DUJOPAS). 9(1b), *https://dx.doi.org/10.4314/dujopas.v9i1b.10*.

Anas, A., Zaharadden, Y. Y., Bagiwa, M. A., Muhammad M. A. (2023c). Scene Change aware Inter-Frame Forgeries Detection Technique for Surveillance Videos Based on Similarities Analysis. *Dutse Journal of Pure and Applied Sciences* (DUJOPAS). 9(1a), *https://dx.doi.org/10.4314/dujopas.v9i1a.3*.

Anas, A., Bagiwa, M. A., Roko A., Buda S., Zaharadden, Y. Y., Bello, A. M., Halimatu, A. A. (2022). An Inter-Frame Forgery Detection Technique for Surveillance Videos Based on Analysis of Similarity. *Sule Lamido Journal of Science and Technology* 4(1), 15-26, *doi.org/1056471/slujst.v4i.265.*

Anwer, F., Aftab, S., Waheed, U., & Muhammad, S. S. (2017)." Agile Software DevelopmentModels TDD, FDD, DSDM, and Crystal Methods: A Survey". International journal of Multidisciplinary sciences and engineering", 8(2), 1-10.

Beck, K., & Gamma, E. (2022). Extreme programming explained: embrace change. Addison wesley professional.

Braude, E. J., & Bernstein, M. E. (2019). Software engineering: modern approaches. Waveland Press.

Conboy, K. (2009). Agility from first principles: Reconstructing the concept of agility in information systems development". Information journal of systems research, 20(3), 329-354. https:// doi.org/10.1287/ijsre.1090.0236.

Jammalamadaka, K., & Krishna, V. R. (2021). Agile software development and challenges. *International Journal of Emerging Technology and Advanced Engineering*, 3(6).

Kashumi M., Rahina M & john G (2023). A Framework For Emotion-Oriented Requirements Change Handling In Agile Software Engineering. *IEEE Transaction on software Engineering, 104-119.*

Schwaber, K., & Beedle, M. (2002). Agile Software Development with Scrum (Vol. 1). Upper Saddle River: Prentice Hall.