American Journal of Software Engineering and Applications
2024, Vol. 12, No. 1, pp. 36-43
https://doi.org/10.11648/j.ajsea.20241201.15

Research Article

Software Testing Framework for the Financial Market

David Ademola Oyemade *

Department of Computer Science, Federal University of Petroleum Resources, Effurun, Nigeria

Abstract

A well designed, developed and tested software is usually reliable and it produces the same consistent outputs for a set of inputs. However, financial markets software is different because it can produce different results for the same periods of back-testing with the same input historical data, usually downloaded from the financial market broker's trading server. These inconsistency of results can confuse a financial market software developer when testing for the profitability of developed expert advisors because a profitable expert advisor can be wrongly discarded as unprofitable, leading to frustrations. This problem can be addressed when new software testing processes and indicators are added to the conventional ones such as functional testing, performance testing, usability testing, etc., associated with normal software development. This paper proposes a software testing framework for the financial market with novel software testing processes and indicators. The proposed software testing framework integrates six software testing processes namely, brokers test, currency pairs test, spread test, weekday-weekend test, back testing-live test and time and space overhead test. The paper further analyzes the problem of time and space overheads associated with the financial market software during back-testing and real life implementation. The framework was applied to real life trading in the Forex financial market. The results show that the proposed framework improves the profitability of the financial market software when applied in different scenarios.

Keywords

Software Testing, Software Development Life Cycle, Framework, Forex, Financial Market, Expert Advisor

1. Introduction

Good software is characterized with qualities such as dependability, reliability and correctness. The outputs of a well designed and developed software product is consistent and predictable with the same set of inputs. This is because conventional and proven methods and tools are applied by the software engineers and other members of the project team. Software products are developed through the application of software architectures, disciplined software processes, software design patterns and software life cycle [1]. Software testing is a prominent phase in the software development life cycle. Software testing ensures that systems functions correctly, meet the stakeholders' needs and deliver the required values to clients. The importance of software testing includes risk mitigation, confidence, compliance, user satisfaction, optimization and cost saving [2]. Software testing types and techniques include functional testing, non-functional testing, capacity testing, usability testing, acceptance testing, regression testing, compliance testing, accessibility testing, configuration testing, unit testing, component testing, integration test, subsystem testing, user testing, performance testing, load testing and security testing, etc. [2-4].

In spite of the availability and application of standard

*Corresponding author: oyeamde.david@fupre.edu.ng (David Ademola Oyemade)

Received: 23 May 2024; **Accepted:** 6 June 2024; **Published:** 19 June 2024

software development methods and the plethora of software testing techniques, the same expert advisors developed for the financial market often produces different results when deployed for automated trading, especially in the Forex financial market. Towards solving this problem, this paper proposes introduction and implementation of a software testing framework consisting of new software testing techniques, indicators and processes.

This paper is structured as follows. The introduction section is followed by the related works in section 2. The materials and method are discussed in section 3. The results and discussion are covered in section 4. The paper ends with the conclusions in section 5.

2. Related Works

In this section, previous works relating to software testing and the financial market are discussed.

Yonathan et al. [5] proposed an approach for the automation of test case for software evaluation based on user requirements and generated from sequence diagram. Casola [6] proposed a methodology to address the concerns of secured software testing in an environment susceptible to cyber-attack. Scommegna [7] proposed a system for testing software architectures, focusing on three tiers architecture to discover faults hidden in the logic layer. Singh et al. [8] proposed the combination of fuzzy analytical hierarchy and fuzzy technique for order preference by similarity for the identification of parameters influencing software testing process. Königa and Kleya [9] focused on production control systems and proposed a software framework for virtual testing and validation using simulation techniques. Byrne et al. [10] conducted a study on software test engineers to determine the effect of their organization behavior on banks' digital transformation. Matalonga et al. [11] focused on the testing of context aware systems and how to validate them in a non-academic environment. Aubertine et al. [12] presented a software reliability assessment tool to achieve test activities defect recovery using Covariate data. Bayramova [13] proposed a neural network approach for the assessment of software reliability. Stradowski and Madeyski [14] explored the challenges of quality assurance in the software testing of Nokia 5G network system, using questionnaires. Bibyan et al. [15] proposed a framework for testing coverage and detecting faults in multi-release software for reliability improvement. Cao et al. [16] proposed the application of Chameleon cluster analysis for fault localization during testing and debugging. Oyemade and Allenotor [17] proposed FAITH software life cycle for Forex financial market and suggested the introduction of new software testing indicators such as the brokers test. Oyemade and Allenotor [18] proposed a quality of service model for Forex MetaTrader platform and demonstrated how service values offered by different brokers affected the profit achievable on the broker's platform.

Some previous works directly relate to the financial market, focusing on systemic risk and price formation [19, 20]. Others focused on the application of fuzzy logic, genetic algorithm, scalability methods, property orientation and greedy algorithm to the forex financial market [21-26]. Other studies proposed the implementation or application of artificial intelligence techniques and machine learning models for profit optimization or for trend prediction [27-30].

Previous studies have focused on different aspects of software testing. No research has considered proposing a software testing framework for the financial market, according to studies. This points to the novelty of this paper and the contribution to knowledge.

3. Materials and Methods

The materials and research methods are provided in this section.

3.1. Materials

The materials used for investigations in this paper include Commercial Network Services virtual private server (VPS), located in the U.S. This was configured with Intel(R) Xeon(R) Platinum 8260 CPU at 2.40 GHz (2 processors) and installed with 2.50 GB RAM. The system type is 64-bit operating system, x64-based processor with Windows Server 2022 Datacenter, version 21H2 and 45 GB hard disk size. Instances of MetaTrader 4, running with Meta Quote programming language were installed on the VPS's operating system. Meta Quote programming language was used for the implementation of trading algorithms and it resembles C programming language. The MetaTrader 4 installations on the VPS include both demo account and real (funded) account. Running the investigations on the virtual private server required a subscription fee of thirty-five U.S. dollars per month.

3.2. Methodology

A thorough literature review of different approaches to software testing was carried out. This was combined with few years of observation of the characteristics of the financial market and its response to the implementation of the software life cycle with expert advisors developed with Meta Quote Language, with particular attention to the drawbacks of the existing and conventional software testing methods. Through these studies and sequel to the studies, a software testing framework for the financial market is proposed.

3.3. The Proposed Framework

The proposed software testing framework consists of six software testing processes namely, brokers test, currency pairs test, spread test, weekday-weekend test, back testing-live test

and time and space overhead test. The conceptual diagram of the proposed model is shown in Figure 1 which graphically illustrates various software testing processes in the framework. The up and down, left and right arrows connecting the various software testing processes depict the synergy of the testing processes and the equal importance of the testing processes. The arrows also indicate that the testing processes can be carried out in any order.

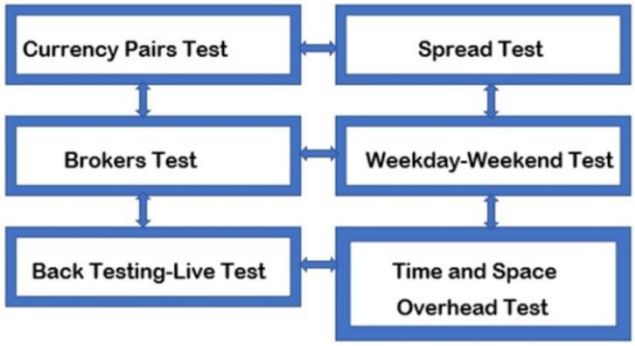

Figure 1. *Conceptual diagram of the proposed software testing framework.*

3.4. Brokers Test

The brokers test is a study of the changing performances and profits of a single expert advisor when implemented with Meta Quote programming language and deployed with MetaTrader 4 on different financial market brokers platform. It is normal to believe that the performances and profits from the various brokers' platform would be very close. However, the reality is far-fetched from what appears to be a normal expectation, emphasizing the significance of this study. The brokers test is adopted from the existing work of Oyemade and Allenotor [18], while the remaining software testing processes can be considered as major extensions of the work. In the existing work, an expert advisor was developed and deployed on three different brokers and made to run for a period of five months. The brokers were represented with three unique code names: FXECN, ICMBR and FXCBR. Figure 2 shows the results of the profits recorded by the different brokers and the wide variances in the values. Figure 2 also shows that FXCBR recorded a loss at the end of the testing period while FXECN recorded a good profit.

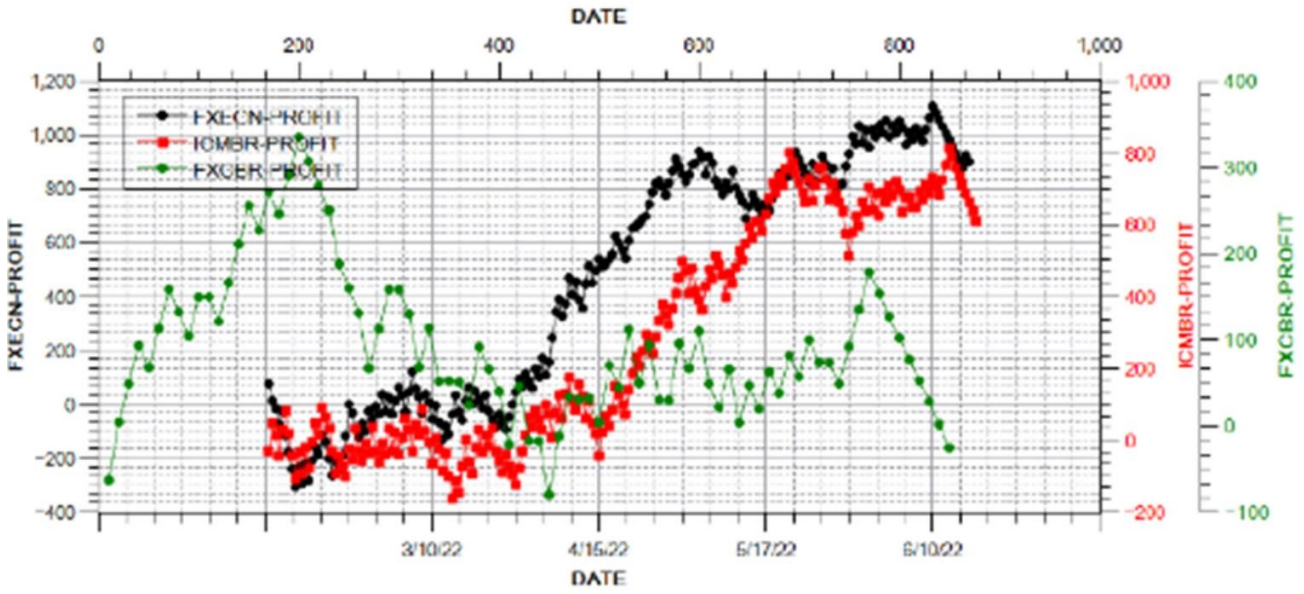

Figure 2. *Brokers Test [18].*

3.5. Currency Pairs Test

The currency pairs test is the process of studying the profitability of an expert advisor when implemented with different currency pairs. Several currency pairs exist, depicting different currencies used across the globe. An example is GBP/USD, denoting Great Britain Pound sterling and United State dollar. Another example is EUR/JPY which denotes Euro and Japanese Yen. Expert advisors can be freely deployed on any of the currency pairs. However, there is need for a software engineer or an expert advisor developer to test the expert advisor with different currency pairs to determine the most profitable. The risk of neglecting this test is that profit made from one currency pairs can be consumed by losses made from another currency pair.

3.6. Spread Test

Spread is the difference between the Ask and the Bid price.

It is believed that the financial market brokers, such as Forex brokers, make part of their gains from the spread. Some brokers use static spread while others adopt dynamic spread. Oblivious to the user, a broker that adopts dynamic spread can set the spread to a very high value during volatility by design, thereby increasing the probability of increased gains by the broker and increased losses by the trader. The effect of using the trading algorithm to set the value of the spread or to use the default spread must be tested by the expert advisor developer. It must be determined whether trading with low spread value can result in lost opportunity while it appears to be a good option. Trading decisions and algorithms should not be based on assumption.

3.7. Weekday-Weekend Test

Financial market software testing using back-testing simulation is time consuming. The testing can take several hours and days depending on the period of the historical data being covered. A way of optimizing testing time is to apply multitasking by running multiple tests using multiple instances of installed MetaTrader application and running different tests on each of them at the same time. However, the limited and expensive computer resources on the VPS makes this impossible. Another way of optimizing testing time is to run the back-testing over the weekend since normal live trading is limited to weekdays. However, the accuracy of the results produced during weekend test can be questionable because volatility is low and spread is usually high in the weekends. Therefore, before weekend test results are used to evaluate and determine the efficiency and profitability of an expert advisor, results obtained during weekday test simulation must be compared with the result obtained during weekend test simulation.

3.8. Back Testing-Live Test

The MetaTrader platform is bundled with a back-testing module which uses simulations to model live trading and it gives developer of expert advisors the privilege of testing their strategies, using historical data. However, the certification of the result produced by the back-testing tool is necessary. The profit or loss attained by an expert advisor during back-testing for a period of time must be compared with the profit attained by the same expert advisor with live trading for the same period of time. This is a way of testing the efficiency, accuracy and reliability of the MetaTrader back-testing module.

3.9. Time and Space Overhead Test

The efficiency of an algorithm is measured by the time and space complexity [31]. However, apart from the run time the expert advisor's algorithm and the space occupied by the expert advisor in the computer memory, back-testing of expert advisors is associated with another problem that can be describe as time and space overheads. During back-testing as well as during normal trading, the ticks, or price data, is continually downloaded into the memory of the client's system. This volume of data continues to grow and can occupy all the hard disk space until the expert advisor becomes irresponsive, forcing the user to terminate the testing process, uninstall and reinstall the expert advisor to free the system's memory for smooth operation. This downloaded, ever-increasing tick data can be referred to as the space overhead during expert advisors' runtime. The several hours it takes to test the expert advisor for a given period of historical data can be described as the time overhead. Thus monitoring of the time and space overheads of the financial market expert advisors during runtime becomes mandatory to avoid inadvertent interruption in software testing and live trading operations.

4. Results and Discussion

The results and discussion of various experiments and simulations of the software testing framework are presented in this section.

4.1. The Result of Currency Pairs Testing

In this investigation the same expert advisor was made to trade on three different currency pairs, GBP/JPY. EUR/JPY and USD/NZD for a period of fourteen weeks. Figure 3 shows the result of the currency pairs testing. It can be seen from Figure 3 that GBP/JPY currency pairs produced the highest profit and should be preferred for trading on a real account for this expert advisor. USD/NZD produced the minimal profit and may be avoided as a choice for real and funded trading account.

Figure 3. *Currency Pairs Test.*

4.2. Analysis of Spread Testing

Table 1 shows six weeks' analysis of spread testing for three different spread values from the same expert advisor which ran for eight weeks with different values of initial or starting spread. It should be clarified that the MetaTrader spread value was set to current option. A constant spread value could be chosen but this was not used for this experiment. The results show that profits are optimized with low values of spread.

Table 1. *Spread test.*

Test Id.	Initial spread value	Number of operations	Number of trades	Profit ($)
1.	70	93	27	-358
2.	10	60	18	-398
3.	13	97	29	-296

4.3. The Result of Weekday-Weekend Testing

Figure 4 shows the result of Weekday-Weekend testing on an expert advisor which ran for a period of six weeks. The chat in Figure 4 clearly shows that weekend testing is not as efficient as weekday trading and should be avoided. It must be mentioned that in real trading on the Forex financial market, trading stops on Friday nights and resumes on Sunday night.

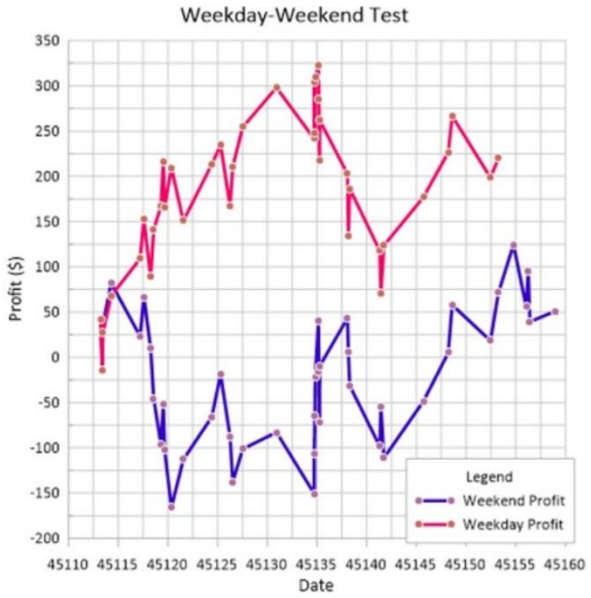

Figure 4. *Weekday-Weekend Test.*

4.4. Results of Back Testing Live Testing

The experiment commenced by deploying two instances of an expert advisor to trade live in the foreign exchange market for a period of eight weeks for two different currency pairs on the same account. GBP/JPY and EUR/JPY were the two currency pairs tested. At the end of the live trading session, back testing of the expert advisor was carried out for the two currency pairs for the same period. The results were them combined on the same csv file for analysis The back testing-live test for GBP JPY currency pair is shown in Figure 5 while back testing live test for EUR JPY currency pair is shown in Figure 6. For the GBP/JPY currency pair, the losses incurred for the live trading was -17 pips while the profit gained for the back testing was 16 pips, giving an error difference of 33 pips.

For the EUR/JPY currency pair, the losses incurred for the live trading was -31 pips while the loss incurred for the back testing was -218 pips, giving an error difference of 249 pips.

This result indicates that the currency pair that produced the minimum error difference should be selected for live trading. In this case considered, GPY/JPY currency pair will be selected.

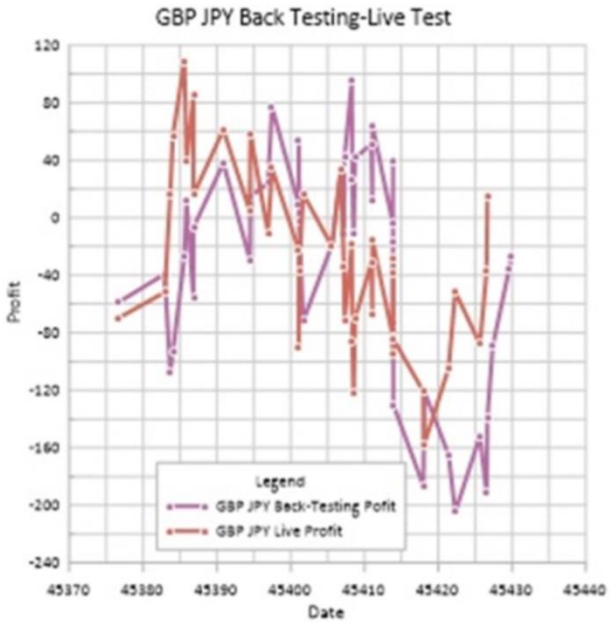

Figure 5. *GBP JPY back testing-live test.*

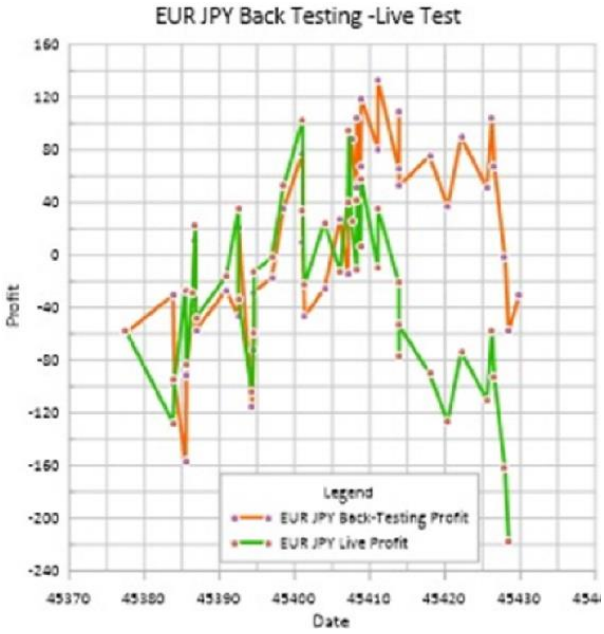

Figure 6. *EUR JPY back testing-live test.*

4.5. Analysis of Time and Space Overhead Testing

Table 2 shows the analysis of time and space overheads of eight weeks back testing for three expert advisors. The expert advisors are identified with the Ids, EXPTC3F, EXPTSENS and EXPTDPF. The analysis shows that only half day software testing using back testing consumed a hard disk space of 7.8 GB. This is a very expensive overhead noting that $35 is charged per month for the VPS employed for the testing and trading. The analysis also emphasizes the importance of code optimization for the expert advisor since it can be seen that a high space overhead does not translate to high profits.

Table 2. *Time and space overheads of eight weeks back-testing for three expert advisors.*

Expert advisor Id.	Profit (pips)	Testing space overhead (GB)	Testing time overhead (hours)
EXPTC3F	155	7.8 GB	12
EXPTSENS	-114	0.7 GB	11.3
EXPTDPF	-25	0.1 GB	11.75

5. Conclusions

Six new software testing processes have been proposed in this paper, all integrated in the proposed software testing framework for the financial market. The empirical results of this paper have shown that the conventional and existing software testing processes and indicators are not sufficient for profitable trading in the financial market. For the currency pairs tested, it has been demonstrated that financial market brokers must be subjected to brokers testing for optimal profit for traders. The results also show that currency pairs must be tested for a trading software to deploy the most profitable for trading with real accounts. The results of the spread testing demonstrate that trading profit is optimal with low spread. Results analysis also show that time and space overheads should be evaluated for the smooth running of the trading software.

Future work shall focus on automated software testing.

Abbreviations

VPS	Virtual Private Server
FAITH	Facts, Analysis, Implementation, Testing, Hope

Acknowledgments

The efforts and time of Mrs. Stella Oyemade who proofread the paper is acknowledged.

Author Contributions

David Ademola Oyemade is the sole author. The author read and approved the final manuscript.

Data Availability Statement

1. The data can be provided upon reasonable request.
2. The data supporting the outcome of this research work has been reported in this manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] Sommerville, I. Engineering Software Products, An Introduction to Modern Software Engineering. Pearson Education Limited, 2021, pp. 1-295.

[2] IEEE Computer Society. The importance of software testing. Available from: https://www.computer.org/resources/importance-of-software-testing?Campaign_ID=263&gad_source=1&gclid=Cj0KCQjwjLGyBhCYARIsAP-qTz1-92gW8Jx3qg1N9ojc-8-w6MpTrm1VE17Yn9a88iq20Wb7m66N0JY4aAjduEALw_wc (accessed 21 May 2024).

[3] Agarwal, B. B., Tayal, S. P., Gupta, M. Software engineering and testing. Jones and Bartlett Publishers, 2010, pp. 161-180.

[4] Mall, R. Fundamentals of Software Engineering, Fourth Edition. PHI Learning Private Limited, 2014, pp. 435-439.

[5] Yonathan, A., Alibasa, M. J., Riskiana, R. R. Generating automated test case from sequence diagram using Pre-order Traversal. Procedia Computer Science, 2024, 234, pp. 1730–1737.

[6] Casola, V., Benedictis, A. D., Mazzocca, C., Orbinato, V. Secure software development and testing: A model-based methodology. Computers & Security, 2024, 137(103639), pp. 1-16. https://doi.org/10.1016/j.cose.2023.103639

[7] Scommegna, L., Verdecchia, R., Vicario, E. Unveiling Faulty User Sequences: A Model-Based Approach to Test Three-Tier Software Architectures. Journal of Systems and Software, 2024, 212(112015), pp. 1-16. https://doi.org/10.1016/j.jss.2024.112015

[8] Singh, V., Kumar, V., Singh, V. B. A hybrid novel fuzzy AHP-TOPSIS technique for selecting parameter-influencing testing in software development. Decision Analytics Journal, 6(100159), 2023, pp. 1-15. https://doi.org/10.1016/j.dajour.2022.100159

[9] Königa, T., Kleya, M. Software framework for virtual testing of a production control system. Procedia Computer Science, 2023, 225, pp. 1495–1503.

[10] Byrne, B., Tuite, A., Organ, J. A Study of the Organizational Behavior of Software Test Engineers, Contributing to the Digital Transformation of Banks in the Irish Financial Sector. IFAC PapersOnLine, 2022, 39, pp. 259-264.

[11] Matalonga, S., Amalfitano, D., Doreste, A., Fasolino, A. R., Travassos, G. H. Alternatives for testing of context-aware software systems in non-academic settings: results from a Rapid Review. Information and Software Technology, 2022, 149(106937), pp. 1-16. https://doi.org/10.1016/j.infsof.2022.106937

[12] Aubertine, J., Kenan Chen, K., Nagaraju, V., Lance Fiondella, L. A covariate software tool to guide test activity allocation. SoftwareX, 2022, 17(100909), pp. 1-6. https://doi.org/10.1016/j.softx.2021.100909

[13] Bayramova, T. A. Development of a Method for Software Reliability Assessment using Neural Networks. Procedia Computer Science, 2023, 230 pp. 445–454.

[14] Stradowski, S., Madeyski, L. Exploring the challenges in software testing of the 5G system at Nokia: A survey. Information and Software Technology, 2023, 153{107067}, pp. 1-18. https://doi.org/10.1016/j.infsof.2022.107067

[15] Bibyan, R., Anand, S., Anu G. Aggarwal, A. G., Gurjeet Kaur, G. Multi-release software model based on testing coverage incorporating random effect (SDE). MethodsX, 2023, 10 (102076), pp. 1-13. https://doi.org/10.1016/j.mex.2023.102076

[16] Cao, H., Chu, Y., Zhao, C., Deng, M. Software multi-fault localization via Chameleon clustering in parallel. Journal of King Saud University - Computer and Information Sciences, 2023, 35(8), pp. 1-10. https://doi.org/10.1016/j.jksuci.2023.101676

[17] Oyemade, D. A., Allenotor, D. FAITH software life cycle model for forex expert advisors. Journal of Advances in Mathematical and Computational Sciences, 9(1), 2021, pp. 1-12.

[18] Oyemade, D. A., Allenotor, D. A quality of service (QoS) model for forex brokers' platforms. International Journal of Innovative Science, Engineering & Technology, 9(6), 2022, pp. 123-132.

[19] Bevilacqua, M., Tunaru, R., Vioto, D. Options-based systemic risk, financial distress, and macroeconomic downturns. Journal of Financial Markets, 65(100834), 2023, pp. 1-35. https://doi.org/10.1016/j.finmar.2023.100834

[20] Bossaerts, F., Yadav, N., Bossaerts, P, Nash C, Todd T et al. Price formation in field prediction markets: The wisdom in the crowd. Journal of Financial Markets 2024; 68(100881): pp. 1-16. https://doi.org/10.1016/j.finmar.2023.100881

[21] Oyemade, D. A., Ekuobase, G. O., Chete, F. O. Fuzzy logic expert advisor topology for foreign exchange market. In Proceedings of the International Conference on Software Engineering and Intelligent Systems, Covenant University.

[22] Oyemade DA, Ojugo AA. An optimized input genetic algorithm model for the financial market. International Journal of Innovative Science, Engineering & Technology, 8(2), 2021 408-419.

[23] Oyemade, D. A., Allenotor, D. A Trade gap scalability model for the forex market. In IEEE 11th International Conference on Ubiquitous Intelligence & Computing and IEEE 11th International Conference on Autonomic & Trusted Computing, Washington, DC, USA; 2014. pp. 867–873. https://doi.org/10.1109/UIC-ATC-ScalCom.2014.38

[24] Oyemade, D, A., Ojugo, A. A. A property oriented pandemic surviving trading model. International Journal of Advanced Trends in Computer Science and Engineering, 9(5), 2020, pp. 7397-7404. https://doi.org/10.30534/ijatcse/2020/71952020

[25] Oyemade, D. A., Enebeli, D. A dynamic level technical indicator model for oil price forecasting. Global Journal of Computer Science and Technology, 21(1), 2021, pp. 5-14. https://doi.org/10.34257/GJCSTGVOL21IS1PG5

[26] Oyemade, D. A. A typified greedy dynamic programming model for the metatrader platform. Journal of Advances in Mathematical and Computational Sciences, 8(3), 2020, pp. 49-60.

[27] Ahmed, S., Hassan, S., Aljohani, N. R., Raheel Nawaz, R. FLF-LSTM: A novel prediction system using Forex Loss Function. Applied Soft Computing, 97, Part B, 2020, pp. 106780. https://doi.org/10.1016/j.asoc.2020.106780

[28] Perla, S., Bisoi, R., Dash, P. K. A hybrid neural network and optimization algorithm for forecasting and trend detection of Forex market indices. Decision Analytics Journal, 6(100193), 2023, pp. 1-16. https://doi.org/10.1016/j.dajour.2023.100193

[29] Ni, L., Li, Y., Wang, X., Zhang, J., Yu, J., Qi, C. Forecasting of Forex Time Series Data Based on Deep Learning. Procedia Computer Science, 147, 2019, pp. 647-652. https://doi.org/10.1016/j.procs.2019.01.189

[30] Almeida, B. J., Neves, R. F., Horta, N. Combining support vector machine with genetic algorithms to optimize investments in forex markets with high leverage. Applied Soft Computing, 64, 2018, pp. 596–613.

[31] Dodmane, R., Aithal, G., Shetty, S. Construction of vector space and its application to facilitate bitwise XOR – Free operation to minimize the time complexity, Journal of King Saud University - Computer and Information Sciences, 34(10), 2022, pp. 9836-9843. https://doi.org/10.1016/j.jksuci.2021.12.015

Biography

David Ademola Oyemade is an Associate Professor of Computer Science at the Federal University of Petroleum Resources, Effurun, Delta State, Nigeria. He holds a PhD degree in Computer Science obtained from the University of Benin, Benin City, Nigeria in 2014. He also holds an M.Sc. degree in Computer Science obtained from the University of Benin, Benin City, Nigeria in 2007 and a postgraduate diploma in Computer Science obtained from the University of Benin, Benin City, in 2004. He is a life member of Nigeria Computer Society (NCS) and a professional member of Association for Computing Machinery (ACM). He has served as a lecturer in the Department of Computer Science, Federal University of Petroleum Resources, Effurun and rose through various ranks. He has supervised several students of at undergraduate and postgraduate levels at the department of Computer Science, Federal University of Petroleum Resources, Effurun. He has many articles in international and local journals. His research area is Software Engineering, Software Architecture, Intelligent Systems and financial market algorithms and modelling.

Research Field

David Ademola Oyemade: Software Engineering, Software Architecture; Intelligent Software Systems, Financial Market Algorithms, Deep Learning, Cloud Computing.