Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 4 Issue 6, June - 2017

Modeling Tax Fraud Security and Control in Edo State: The UML Approach

Oshoiribhor E. O.
Dept. of Computer Science,
Ambrose Alli University,
Ekpoma, Nigeria
emmaoshor2001@gmail.com

Ojieabu C. E.
Dept. of Elect/Elect Eng.
Ambrose Alli University,
Ekpoma, Nigeria
bishopeghosa@yahoo.ca

John-Otumu A. M.
ICT Directorate,
Ambrose Alli University,
Ekpoma, Nigeria
macgregor.otumu@gmail.com

***Abstract*—The unstructured method of apportioning tax levies to tax payers of various businesses in Edo state has led to over-estimating or under-estimating the tax amount charged; thereby causing the rampant fraudulent activity of cash suppression and diversion in the tax collection system. This scenario motivated the research interest in order to control and secure fraud in Edo state tax collection system. The UML interaction approach was used to critically model the security and control, alongside software agent designed to secure and control tax fraud.**

Keywords—Modeling, Tax, Fraud, Software agent, Security, UML.

I. INTRODUCTION

Fraudulent act is now one of the ways of life; fraud can never stop unless man ceases to exist, but can be detected and reduced to some extent [1]. The world is overwhelmed with millions of inexpensive gigabyte disks containing terabytes of data [2]. From research findings, It is estimated that these data stored in all corporate and government databases all over the world doubles every fifteen to twenty months.

However, this has resulted in a data rich but information poor situation where there is a widening gap between the explosive growth of data and its types, and the ability to analyze and interpret it. Hence there is a need for a new generation of automated and intelligent tools and techniques [3], known as investigative data mining, to look for patterns in data. Data mining is the means used in extracting hidden knowledge from a data set; this would be knowledge that is not readily obtained by traditional means such as queries or statistical analysis [4]. Hidden knowledge of data can be used for classification and estimation of new instances and for prediction of future events [4].

An introduction and critical analysis to UML diagrams and notations is presented in this section. Merits and de-merits of UML are listed. The reasoning of linking and formalizing UML is provided. UML has various benefits for modeling of complex systems. For example, UML is a semi-formal language in which each element of the language is strongly defined [5]. That is you are confident when modeling a particular facet of a system in a sense that it will not mislead to an incorrect design. UML is a concise and easy to under-stand designing language [6]. The entire language is made up of simple and straightforward concepts and notations. It is comprehensive language and describes all important aspect of a system. Although UML is not a formal language but it has enough expressive power to handle massive and complex systems [7]. It is the result of best practices in modeling of complex systems using object-oriented concepts and has proved to be a successful modeling practice. UML has become a de facto Standard for modeling of systems using object oriented technology [8]. Further discussion about UML analysis can be found in [9]. Despite the above benefits, for example, UML lacks formal semantics. Meanings are hidden under diagrams which create ambiguities at the implementations level. That is why integration of UML with other appropriate approaches is required for the complete and consistent modeling.

II. LITERATURE REVIEW

However, this paper describes the software tools but not the actual documentation security processes, control and organization. System documentation can be structured using a modeling language [10]. A modeling language can be any artificial language which

could be used in expressing knowledge in a structure that is defined by a consistent set of rules. The rules are used to interpret the meaning of the different components in the structure [11].

UML (Unified Modeling Language) [12] is one of the most widely used languages for defining and specifying document systems [13]. To model systems with certain specific needs, UML can be extended in two different ways such as:

- The UML extension by means of a profile providing the stereotypes, tagged values and constraints required to specify the peculiarities of the modeled system.
- Extension of the Meta Object Facility (MOF) [14], by way of modeling languages from which UML is defined. UML extensions have also been defined for:
- Documenting web site development [15]
- Building web-based remote monitoring and fault diagnosis systems [16]

Although there exists a lot of work [17] on integration of approaches but there does not exists much work on linking UML diagrams with formal approaches. This is because the hidden semantics under the UML diagrams cannot be transformed easily into formal notations. It is mentioned that only closely related work is discussed in this section. For example, [18] has developed Alloy Constraint Analyzer tool supporting the description of a system whose state space involves relational structures which are complex in nature. By the tool it is possible to analyze and develop a model by investigating the consequences of given constraints by an incremental approach. A case study is discussed by a formal verification method for Cooperative composition Modeling Language (CCML) in [19]. Information is captured using sequence and use case diagrams as a functional model by taking a case study. A method for translating and verifying UML sequence diagrams to Petri nets for deadlock, safety and liveness properties by model checking is discussed in [20]. It is investigated that reliability issues using fuzzy logic and Petri nets in [21]. Formalization of the UML is proposed by focusing on basic constructs of class structures by taking simple case studies in [22]. A comparison of UML, state-charts, Z notation, Petri nets and fuzzy logic is presented by taking a simple case study on commerce system as discussed in [23]. Some other relevant work is also seen in [24].

III. METHODOLOGY

The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components.

In this paper, we made use of two types of UML diagrams to represent the application graphically:

1. Use Case Diagram
2. Sequence Diagram

Use Case Diagram

The use case diagram describes what a system does from the standpoint of an external observer. The emphasis is on what a system does rather than how is does it. Use case diagrams are closely connected to scenarios.

The following practical scenarios clearly demonstrate what happens when a user interacts with the system features.

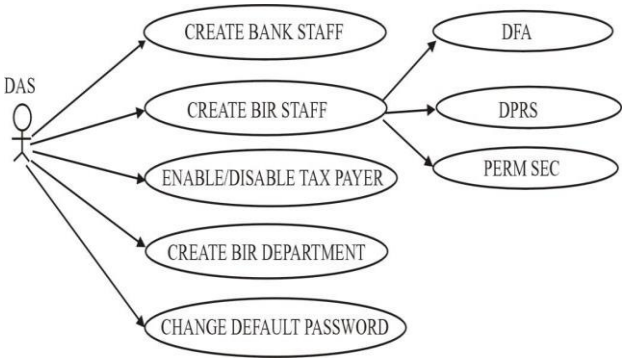

Fig. 1: Modeling DAS interaction with system features

Fig. 1 shows the interaction between the Director of Administration and Supply (DAS) with the system features; the DAS can create other staff users like the DFA, DPRS, PERM SEC and bank users, enable/disable tax payers and change its default password.

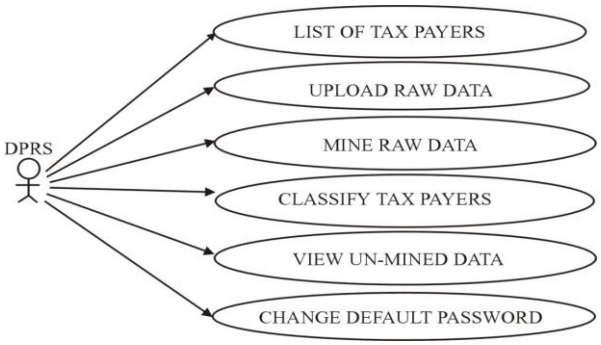

Fig. 2: Modeling DPRS interaction between with system features

Fig. 2 depicts the modeling of the Director of Planning, Research and Statistics (DPRS) and as object and how the object interacts with the system features. The DPRS can view list of tax payers, uploads and mines raw data for the DFA, who performs the profitability analysis. The DPRS can also perform the operation of classifying tax payers into well-structured tiers, view un-mined data and change its default password.

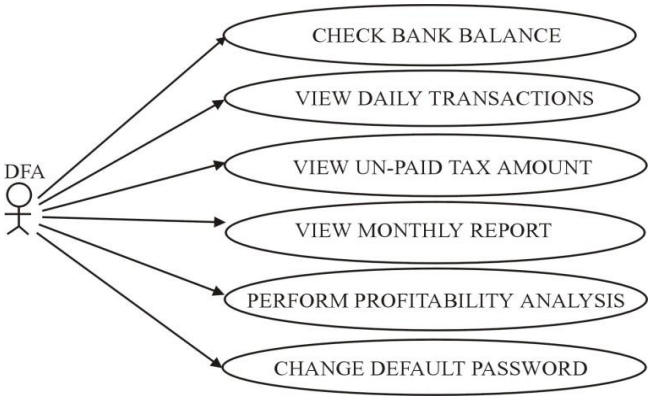

Fig. 3: Modeling DFA interaction with system features

Fig. 3 shows the modeling of the Director of Finance and Accounts (DFA) interaction with system features. The DFAs responsibility is to check bank balance, view daily transactions, view un-paid tax amount, view monthly report, compute the profitability analysis and change default password.

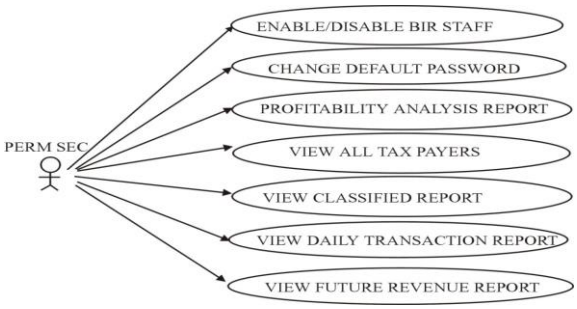

Fig. 4: Modeling the Permanent secretary's interaction with system features

Fig. 4 shows how the permanent secretary views and interacts with the system features as and when required.

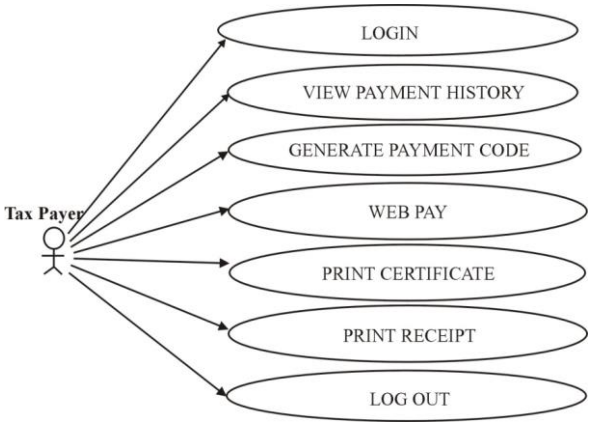

Fig. 5: Modeling the Tax payers' interaction with system features

Fig. 5 however, demonstrates the various activities of the tax payers such as generating payment code, make web pay, print certificate and also view payment history.

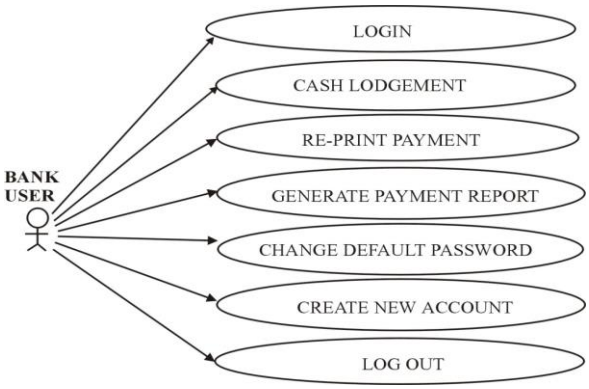

Fig. 6: Modeling the bank staff user's interaction with system features

Fig. 6 nevertheless, demonstrates the function of the Bank staff user with the system features.

Sequence Diagram

A sequence diagram is an interaction diagram that details how operations are carried out, what messages are sent and when it is sent. Sequence diagrams are organized according to time. The time progresses as you go down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence.

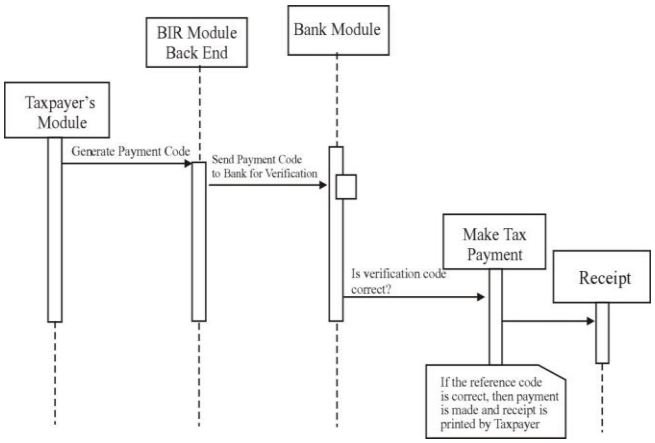

Fig. 7: Modeling the process for making tax payment

Fig. 7 shows the sequence of interactions between the tax payer with the BIR and Bank modules in order to make tax payment. The taxpayer module sends generate payment code message to BIR Module. The BIR Module's Agent called the "BIRGENT" then sends the generated reference code to the taxpayer through the taxpayer module. The taxpayer then, proceeds to Bank module for payment. The Bank Module checks if the reference code presented by the taxpayer is genuine and correct, if genuine/correct, the bank module will send 'make payment()' message and a receipt is printed for the taxpayer.

Each vertical dotted line is a lifeline, representing the time that an object exists. Each arrow is a message call. An arrow goes from the sender to the top of the activation bar of the message on the receiver's lifeline. The activation bar represents the duration of execution of the message.

In Fig. 7 the Bank Module issues a self-call to determine if the reference code is genuine/correct/exist. If so, then the Bank Module receives the money and a receipt is printed for the taxpayer.

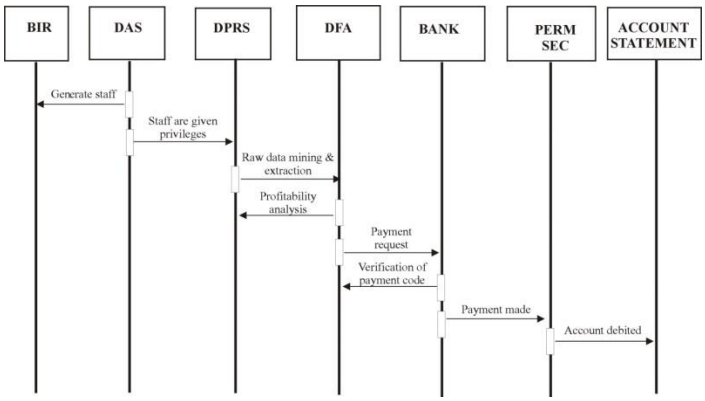

Fig. 8: Modeling the Tax Fraud Security System

Fig. 8 demonstrates clearly the security aspect using the sequence diagram. The taxpayer makes payment request via the Board of internal revenue. The BIR, in turn generates unique reference code, which must be confirmed unique by the agent before payment can be made by the taxpayer. The bank module verifies the reference code before the tax pay can eventually login for his/her payment receipt, which is generated by the BIR.

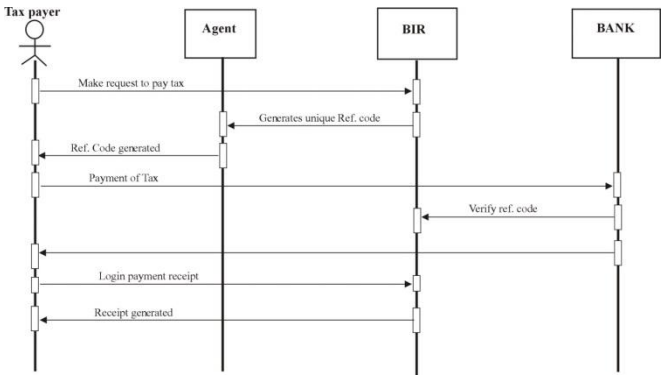

Fig. 9: Modeling the Tax Fraud Control System

Fig. 9 demonstrates how Admin staffs are created by the BIR, who in turns create the DAS i.e. Director of Admin and supply. The DAS then create other staff with privileges. The DPRS performs the mining. Thereafter, the DFA performs the profitability analyses and sent to

the DPRS again for extraction. Payment request is now made to the bank, who in turn verifies the payment code for correction. At this point the permanent secretary is able to view the account details and later report to the state governor.

IV. DISCUSSION

The application of UML is a practical demonstration what happens when a user interacts with the system features. The system clearly shows the interaction between

i. The Director Admin and Supply (DAS), who is responsible for creating other Staff.
ii. The Director of Personnel Research and Statistics (DPRS) updates and mines raw data which is therefore sent to the Director of Finance and Accounting, (DFA) who does the profitability Analysis of the tax payer.
iii. The DPRS performs the classification and extraction.

Lastly, the Permanent Secretary is able to view from his desk all of these activations already demonstrated, including the results, who in turn is responsible to the State Government.

V. CONCLUSION

This paper addresses the rampant problem of cash suppression and diversion caused by the manual and semi-automated operations in the existing tax collection system of Edo State. The problems identified from the works of some research scholars in the field of data mining and artificial intelligence for security and fraud detection in different areas has motivated our interest in this area of research in order to ensure maximum control and security in the tax collection system.

The Unified Modeling Language (UML) is a powerful, object-oriented and visualized system analysis and modeling language. It uses a set of complex modeling techniques and is widely applied to various areas. The use of UML-based object-oriented visual modeling can provide the software developers a unified, flexible and understandable representation model, which can reduce uncertainty in the system design and it is more conducive to the expansion and test of the system. This paper uses the Unified Modeling Language to analyze and design the fraud security and control system. We can see that, as a modeling language in software engineering, UML represents the development direction of the object-oriented method in software development technology, which has a very important economic value and a very good application prospect.

REFERENCES

[1] Abagnale, F. W. (2001). *The Art of the Steal: How to Protect Yourself and Your Business from Fraud.* Penguin Random House, Australia, ISBN: 1742747221.
[2] Phua, C., Alahakoon, D. & Lee, V. (2004). Minority Report in Fraud Detection: Classification of Skewed Data, *SIGKDD Explorations* **6**(1): 50-59.
[3] Fayyad, U., Piatesky-Shapiro, G., Smyth, P., and Uthurusamy, R. (1996) Advances in Knowledge Discovery and Data Mining, AAAI Press, CA, USA.
[4] Roiger, R. J., and Geatz, M. W. (2003). A Tutorial Based Primer, Addison Wesley, U.S.A
[5] Borges R. and Mota, A. (2003). Integrating UML and Formal Methods, Electronic Notes in Theoretical Computer Science, Vol. 184, 2003, pp. 97-112. doi:10.1016/j.entcs.2007.03.017
[6] Podeswa, H. (2009). "UML for IT Business Analyst," 2nd Edition, Course Technology,
[7] Dennis, A., Wixom, B. H., and Tegarden, D. (2005). "Systems Analysis and Design with UML," 3rd Edition, Wiley, Hoboken.
[8] Miles, R., and Hamilton, K. (2006). Learning UML 2.0. O'Reilly Media.
[9] Zarina, S., Alias, N., Halip, M. M., and Idrus, B. (2006). "Formal Specification and Validation of Selective Acknowledgement Protocol Using Z/EVES Theorem Prover," Journal of Applied Sciences, Vol. 6, No. 8, 2006, pp. 1712-1719. doi:10.3923/jas.2006.1712.1719.
[10] Jackson, D., Schechter, I., and Shlyakhter, I. (2000). "Alcoa: The Alloy Constraint Analyzer. In Proceedings of International Conference on Software Engineering, Limerick, 4-11 June 2000, pp. 730-733. doi:10.1109/ICSE. 2000.870482
[11] Moeini, A., and Mesbah, R. O. (2009). "Specification and Development of Database Applications Based on Z and SQL" In Proceedings of International Conference on Information Management and Engineering, Kuala Lumpur, 3-5 April 2009, pp. 399-405. doi:10.1109/ICIME.2009.143
[12] Zhang, X. G. and Liu, H. (2011). "Formal Verification for CCML Based Web Service Composition," Information Technology Journal,

Vol. 10, No. 9, 2011, pp. 1692-1700. doi:10.3923/itj.2011.1692.1700

[13] Kim, S. K. and Carrington, D. (2000). "An Integrated Framework with UML and Object-Z for Developing a Precise and Understandable Specification: The Light Control Case Study," Proceedings of 7th Asia-Pacific Software Engineering Conference (APSEC), 5-8 December 2000, pp. 240-248. doi:10.1109/APSEC.2000.896705

[14] Heiner, M., and Heisel, M. (1999). "Modeling Safety Critical Systems with Z and Petri-Nets," Proceedings of International Conference on Computer Safety, Reliability and Security, Toulouse, 27-29 September 1999, pp. 361-374.

[15] Cunha, E., Custodio, M., Rocha, H., and Barreto, R. (2011). "Formal Verification of UML Sequence Diagrams in the Embedded Systems Context," Brazilian Symposium on Computing System Engineering (SBESC), 2011, pp. 39-45.

[16] Leading, H. and Souquieres, J. (2002). "Integration of UML and B Specification Techniques: Systematic Transformation from OCL Expressions into B," Proceedings of 9th Asia-Pacific Software Engineering Conference, Gold Coast, 4-6 December 2002, p. 495.

[17] Raymond, T. B. (2004). "Integrating Formal Methods by Unifying Abstractions," Springer, Berlin, 2004, pp. 441-460.

[18] Shi, Z. (2012). "Intelligent Target Fusion Recognition Based on Fuzzy Petri Nets," Information Technology Journal, Vol. 11, No. 4, 2012, pp. 500-503. doi:10.3923/itj.2012.500.503

[19] Ma, Z. M. (2008). "Fuzzy Conceptual Information Modeling in UML Data Model". In International Symposium on Computer Science and Computational Technology. Shanghai 20-22 December 2008, pp. 331-334. doi:10.1109/ISCSCT.2008.353

[20] Ehikioya, S. A. and Ola, B. (2004). "A Comparison of Formalisms for Electronic Commerce Systems," Proceedings of International Conference on Computational Cybernetics, Vienna, 30 August-1 September 2004, pp. 253-258. doi:10.1109/ICCCYB. 2004.1437721

[21] Changchien, W. S., Shen, J. J., and Lin, T. Y. (2002). "A Preliminary Correctness Evaluation Model of Object-Oriented Software Based on UML," Journal of Applied Sciences, Vol. 2, No. 3, 2002, pp. 356-365. doi:10.3923/jas.2002.356.365.