* 1. **Field Trial Evaluation**

The performance and perception evaluation of this framework was done using Performance Appraisal Form (PAF) and Technology Acceptance Model (TAM). TAM, according to [27] is widely used as one of the prominent information system theories gives a better understanding of users’ acceptance of information systems or information technology (IT). Ten software experts were given PAF and perception form to assess this framework with an extensive interview. For PAF, six specialized indicators were chosen to measure the success rate of software project which were: cost, time quality, scope, complexity and functional requirement. PAF used special rating against these indicators which were: “5 – Exceptional”, “4 –Exceeds Expectation”, “3 – Meet Expectation”, “2 – Below Expectation” and “1 – Need Expectation” according to [18]. Likewise for perception form, nine construct were to describe Perceived Case of Use (PEOU) and Perceived Usefulness (PU). The same scale of grading used for PAF was equally used for perception evaluation.

The Performance Evaluation (PE) score is obtained by dividing the summation of individual scores by number of participant as shown in equation 6 below. Analysis of these results was compared with Evaluation concepts scale [28]. Performance and perception evaluation carried out by the ten experts were shown in table 5 and 6 respectively.

PE (6)

Evaluation concepts scale judges based on the quality of performance. The labels used were: “outstanding”, “Good”, “satisfactory”, “marginal” and “poor”. Although, there are other evaluation scales such as Behavioural frequency scale and standard scale; evaluation concept scale has been found to be suited for this work. Table 4 shows the evaluation concept scale. Overall score was given in this table alongside with the interpretation.

PAF analysis was carried out by finding the average of each indicator for the selected ten experts and the final result was compared with evaluation concept scale to finalise the performance evaluation carried out as seen in table 5 below.

Perception evaluation analysis was carried out by finding the average of each indicator for the selected ten experts and the final result was compared with evaluation concept scale to finalise the perception evaluation result as seen in table 6 below.

1. **Result & Discussion**

The problem with the success of software project is the lack of proactive risk management procedure. This work implemented the proposed ontology-based risk management framework for software project through software development lifecycle on ten different software projects. Protégé OWL was used to build the software risk ontology knowledge repository while ontology consistence was checked with the aid of FACT++ reasoner which was also used to submit DL query.

The approach used in this study presented risks at each software development lifecycle phases. The performance and perception evaluation of this study was carried out using Performance Appraisal form (PAF) and Technology Acceptance Model (TAM) with extensive interview with the software experts.

The outcomes of these evaluations were compared with the Evaluation Concept Scale (ECS) and the following deductions were made:

1. For performance evaluation, when the result outcome was compared with the Evaluation Concept scale, it was discovered that: cost has a mean value of 3.20 which means that it has rating “Good”. This shows that no additional cost was incurred during the projects execution used to test the ORM framework and this could be seen from the assessment of the experts. For Duration (time) and Scope, the mean grading were the same as that of cost which is 3.20. Looking through the PAF assessment according to table 5, all the experts scored duration 3 except for two expert that scored 4 which shows that the project meet up with the target and were delivered on time. Meanwhile, only one expert scored 2 for scope while others experts scored 4 and 3; yet the delivered system were able to perform within specific description of what the end result should be or accomplish. Time and cost do have impact on the quality of a project, but as it can be seen here, the fact that these projects were delivered on time and at a budgeted cost does not have a negative impact on the quality of the delivered system. This was seen from the table 6, quality mean score was 3.30 which were higher than that of cost and time; this shows that projects delivered were of a high quality. Finally, the functional requirement as an indicator equally scored the same value as quality. This implies that, the system were able to perform all the specified functions.
2. For perception evaluation, when the result outcome in table 8 was compared with the Evaluation Concept scale, it was discovered that all the nine constructs exceed 3.0 except risk documentation which is 2.90 telling us that more risk factors will be added to this framework in the future. Meanwhile, means of updating software risk ontology has been provided in the course of this work. Portability meet up with exceptional rating telling us that the framework can be used on any platform and also supportability was rated 3.9 which implies that the software experts found this framework useful in their day-to-day activities and they gave their intention of using the framework in the course of their work. Understandability means score was 3.3 which means the framework was comprehensive enough and can be easily interpreted by all. Also, usefulness and changeability construct scores were 3.5 which means, software experts found this framework to be an essential tools in achieving goals in the course of their duties. Accessibility as a construct defends the importance of building the ORM framework and this shows from the mean score, 3.6, the framework was easily accessed by all the ten experts. Buildability asked the question on how straight forward has this framework helps in meeting pre- requisite for developing a successful software project? The mean score 3.1 was able to answer this question because the score exceeded the satisfactory level which means this framework is straight forward in guiding professionals in getting all their requirement rights. This answer the third research questions of this study.

In lieu of the aforementioned, the following suggestions were made.

1. This framework could help experts to avoid re-work which usually cause delay in the delivery of project, budget extension and even, the poor quality of software project
2. ORM framework for software project could help in converting tacit knowledge to explicit knowledge which may be used as a reference by software developers in the future.
3. This framework being ontology-based will promote knowledge sharing and reusability in software development domain
4. **Threat to validity**

Threats to validity are many but, the four categories discussed in [29] are conclusion, internal, construct and external validity. These four shall be discussed in relative to this work.

* 1. Conclusion validity

Conclusion validity is mainly concerned with the ability to infer a rightful conclusion about relationship between treatment and outcome. In our work, conclusion validity deals with the relationship between the tools used for risk identification and the outcome in terms of number of risks identified at various phases of software lifecycle or the evaluation responses by the experts. The important factor that needs to be considered is whether the sample size is large enough to justify the conclusion drawn from this work.

The effect claimed from performance evaluation data was a significant advantage for RM framework in cost reduction, with an effect size of 3.20. α is used to denote Type I error while β denote Type II error, the following relationship holds for equations 7 to 9:

[30] (7)

If we use and 0.20

The standard normal deviate for = 1.960

The standard normal deviate for = 0.842

(8)

= 4.080

(9)

= 7.849

The effect size for this work is 3.20,

= 3.13

Yields N = 3.13 which means minimum of 3 experts would be needed to perform the evaluation [30]. Since 10 experts were involved in this regard, we have enough number of observations for our conclusion.

* 1. Internal validity

Internal validity deals with relationship between treatment and outcome. It assesses whether the observed outcome were due to the treatment or to other factors. Some of the threats to internal validity are:

6.2.1 Expert selection bias is possible if one expert is cleverer than the other. This threat was countered by our experimental design in which workshop was organised for software experts purposely for risk identification and questionnaire were distributed to them for assessment under the same time limits and other conditions. The risks were jointly identified and categorised by the experts. Also, some risks were wed out by the experts.

The research work might have caused invalid conclusion if experts have had previous knowledge about the usage of this framework rather, the operation of the framework done by the researcher to the experts was the knowledge available to them. The outcome of the performance evaluation shows that the framework is generally acceptable by the experts.

6.2.2 Bias in tutorials would occur if the framework was not presented in a manner easily operable. This framework is portable and it is presented in a format that is understandable by the experts. The researcher took time to demonstrate the operation of the framework to the expert in order to enhance the easy operation.

6.2.3 Training and task time

Training time is short and it took 15minutes to familiarize the expert with the protégé editor. The fact that the participants were all software experts made this possible. Task time took 18working days. Since the framework serves a guide to building a successful software project. This means the task time and project duration is the same. At every phase of project development, the framework needs to be considered to check for risks that must be guided against.

6.2.4 Researcher expectation would occurred if the experts had perceived that RM framework was developed by student and that positively assessment of the framework will help in meeting up with time limit. This threat is too complex to control but, there was no mention, neither orally nor in action that the research must be submitted within a time frame.

* 1. Construct validity

Construct validity concerns whether it is acceptable to draw from the observation made in the experiment to the theoretical constructs that the researcher was trying to observe. Based on performance data, we are trying to observe the effectiveness and coverage of the framework, and this was measured in terms of scores allotted by experts to software success selective indicators. Hence, two threats were considered.

6.3.1 Was the use of technique in the experiment an ideal representative of the theoretical techniques investigated (qualitative risk analysis using AHP). This work overcame this threat by using multiple techniques in identifying software risks. Workshop (brainstorming session), use of questionnaire, organisational documents and review of literature were used to gather the risks. The uses of multiple techniques ensure that no risk is left behind.

6.3.2 Were the measures used such as evaluation concept scale a representative of the metrics they were measured with for effectiveness in software risk management framework? There are other evaluation scales such as behavioural frequency scale and standard scale that can be used to score the output of this work. Since the effectiveness of this framework is the measure of success of this research, the two above mentioned metrics were unsuitable for the measurement.

* 1. External validity

External validity is concerned with generalization – whether there is a possibility of generalizing from the experimental setting to industrial system practice. The major issue here is, can we generalize our results to the software industry?

Some of the threats to consider are:

6.4.1 The use of students instead of practitioners

According to [31], the factor that determine subject performance in research is not students versus practitioners rather, their competence level. However, this threat does not affect this work because software experts having a minimum of 7years experience where engage in this research.

6.4.2 Lack of motivation

In an experiment, when students know that they stand not to gain or loss anything, the interest might be lost. In this work, participants were highly motivated because they saw the frame work has a potent tool in their day-to-day activities in software development.

1. **Conclusion**

This framework verifies how software project risk ontology could improve the on-time delivery of quality software project using case studies demonstration and equally reduce the impacts of risks on the software project being undertaken. The contacted software experts were able to incorporate knowledge reuse in risk management through the implementation of ORM framework and software risk ontology development model.

The proposed software risk ontology was developed using literature review, expert interview. The use of case studies to validate this study proved that this framework could assist the software developers to conduct knowledge reuse during risk management procedures to reduce risks pose to software projects. Moreover, the use of explicit knowledge through acquisition of tacit knowledge could be applied to software development workflow and thereby leading to high performance in software development lifecycle. The conversion of tactic knowledge to explicit knowledge could lead to knowledge reuse.

In conclusion, ORM framework for software project could be applied to other software development because the model proposed here was with reference to general software risk management standard procedure. Likewise, this framework was validated through case studies with extensive experts interview using PAF and the perception evaluation was carried out using TAM.

This framework is therefore recommended to software developers so as to meet up with all the pre-requisite in the delivery of high performance software projects. For the future research we hope to carry out software projects risk management for the cloud.

**References**

[1] MCA (Sem –IV) Paper – Software Project management – Reader, 25-28 (accessed 26.01.16).

[2] Chapman, C., Project risk analysis and management – PRAM the generic process, International Journal of Project Management. 15(5): (1997) 273 – 281.

[3] M. Boban, Z. Pozgaj H. Seric, Strategic for successful software development risk management, 8 (2003) 77-91.

[4] D. Brandon, Project management for modern information system, Idea Group Inc (IGI), 6 (2006) 417.

[5] A. Surbhi, C. Vinay, Decision support system for software risk analysis during software development, International Journal for Science and Emerging Technologies with Latest Trends. 2(1): (2012) 29-35.

[6] Department of Energy Quality Managers, Software Quality Assurance Subcommittee. Reference Document SQAS21.01.00 – 1999 (accessed 26.01.16).

[7] D. Neef, Making the case for knowledge management: the bigger picture Knowledge and Process Management, 6(2): (1999) 101-112.

[8] R. Tenkasi, R. Boland, Exploring knowledge diversity in knowledge Intensive Firms: a new role for information system, Journal of Organisational Change Management. 9(1): (1996) 79-91.

[9] S. Raub, C.C. Ruling, The Knowledge management tussle- speech communities and rhetorical strategies in the development of knowledge management, Journal of Information Technology. 16(2): (2001) 113-130,

[10] H. Scarbrough, Knowledge Management, HRM and the Innovation Process, International Journal of Manpower. 24(5): (2003) 501-516.

[11] M. Polanyi, The tactic dimension, Annals of the American Academy, AAPSS. 458 (1996)

[12] Afolabi I.,Daramola O. and Adio T**.**., Developing Domain Ontology for Nigerian History. Aust. J. basic & Appl. Sc., 8(4): (2014) 1-3

[13]Ricardo de Almeida Falbo, Ana Candida Cruz Natali, Paula Gomes Mian, Gleidson Bertollo, Fabiano Borges Ruy, ODE: Ontology-based software development environment, CACIC– RedUNCI. (2003) 1124 -1135.

[14] H. Ping Tserng, Samuel Y.L. Yin, R.J. Dzeng, B. Wou, M.D. Tsai, W.Y. Chen, A study of Ontology-based Risk Management framework of Construction Projects through Project Lifecycle, automation in construction, Elsevier Journal. 18 (2009) 994 – 1008.

[15] C.R. Rene Robin, G.V. Uma, Design and development of ontology for risk management in software project management, International Symposium on Computing, Communication, and Control (ISCCC 2009) Proc .of CSIT. 1 (2011) 253-254.

[16] H. Haneen, A. Shihadeh, M. Hasan, K. Thair, Risk factors in software development phases, European Scientific Journal. 10, (3): (2014) 1857–7431.

[17] Antonio de Amescua, José García, Manuel Velasco, Paloma Martínez, Belén Ruiz, Juan Llorens, Luis García, Antonio Calvo-manzano, Tomás San Feliu, A software project management framework, Information Systems Management Journal. 21:2 (2004) 78-85.

[18] J. Dean, B. Trevor, V. Pepijn, Methodologies for ontology development, department of Computer Science, University of Liverpool, UK., L697ZF

[19] Natalya F. Noy, Deborah L. McGuinness, Ontology development 101: A Guide to Creating Your First Ontology, Stanford University, Stanford, CA, 94305

[20] S. Boyce, C. Pahl, Developing Domain Ontologies for Course Content, In *Educational Technology & Society*. 10 (3) 275-288.

[21] A. Norris Syed, I. Marta, S. Shazia, Compliance management ontology – a shared conceptualization for research and practice in compliance management, Inf Syst Front DOI 10.1007/s10796-016-9631-4, 2016

[22] J. F. Sowa, Knowledge Representation – Logical, Philosophical, and Computational Foundations, Pacific Grove, CA, USA: Brooks/Cole, 2000.

[23] Department of Energy Quality Managers Software Quality Assurance Subcommittee Reference Document SQAS21.01.00 – 1999, February 2000.

[24] T. L. Saaty, The analytic hierarchy process, New York, NY: McGraw-Hill, 2000, pp. 45-49

[25] E. H. Forman, T. L. Saaty, Expert choice. Pittsburgh, PA: Expert Choice, 1983

[26] Web Ontology Language (OWL), <http://www.w3.org/2015/sw/WebOnt/>

[27] V. Venkatesh, H. Bala, Technology acceptance model 3 and a research agenda on interventions, Decision Sciences, 39, 2008,273 –315. doi:10.1111/j.1540-5915.2008.00192.x.

[28] K. Tanjila, John Grundy, Performance Appraisal of Software Testers, Information & Software Technology. (2013) 5-12.

[29] C. Wohlin, P. Runeson, M. Host, M.C. Ohlsson, B. Regnell, A. Wesslen. Experimentation in Software Engineering: An introduction. Kluwer Academic, Norwell, MA, USA, 2000.

[30] Chow S-C, Shao J, Wang H. Sample size calculations in clinical research. 2nd ed. Boca Raton: Chapman & Hall/CRC; 2008. Section 3.2.1, page 58.

[31] E. Arisholm, D.I.K. Sjoberg. Evaluating the effect of a delegated versus centralized control style on maintainability of object-oriented software, IEEE Transactions on Software Engineering 30 (2004) 521-534.

**Tables**

|  |  |
| --- | --- |
| **Risk classes** | **CR (%)** |
| Requirement analysis | 3.42% |
| Design | 2.90% |
| Implementation & Unit testing | 0% |
| Integration & System Testing | 2.02% |
| Operation & Maintenance | 3.88% |

Table 1: Consistency Ratio of each risk class of the selected experts

What risks factors affect installation and operation subcategories?

DL Query*: operation or installation or has\_effect min2*

***Result:***

***Equivalent classes*** *(0)*

***Subclasses*** *(2)*

*Installation*

*operation*

***Descendant classes*** *(2)*

*Installation*

*Operation*

***instances (****5)*

*Problems\_in\_installation*

*Environmental\_change*

*Difficulties\_in\_using\_the\_system*

*Emergence\_of\_new\_requirement*

*The\_effect\_on\_the \_environment*

**Table 2: DL query and its response**

Under the design specification, what risk factors are present?

*DL Query: Design\_Specification and has\_effect self*

***Result:***

***Equivalent classes*** *(0)*

***Subclasses*** *(0)*

***Descendant classes*** *(0)*

***Instances*** *(3)*

*Extensive\_specification*

*Difficulties\_in\_allocating\_functions\_to\_components*

*Omitting\_data\_processing\_functions*

**Table 3: DL query and its response**

|  |  |
| --- | --- |
| **Overall Score** | **Interpretation** |
| 0 – 0.99 | Poor |
| 1 – 1.99 | Marginal |
| 2 -2.99 | Satisfactory |
| 3 - 3.99 | Good |
| 4 – 5 | Outstanding |

**Table 4: Evaluation Concept Scale**

Cost Duration Complexity Quality Scope Functional Requirement

2 3 4 3 3 3

4 3 3 4 3 3

4 3 2 2 4 3

4 3 4 3 3 3

3 3 2 3 3 4

3 3 3 5 3 4

3 4 2 4 4 4

4 3 3 3 3 3

2 4 3 3 4 3

3 3 3 3 2 4

Mean 3.20 3.20 2.90 3.30 3.20 3.30

Table 5: Summary of Software Expert Evaluation Using Selected Indicators with mean value

UA RD BA UF AC PA SA US CA

4 3 3 4 4 4 3 3 3

3 3 3 2 3 4 4 4 3

4 3 3 4 3 2 3 3 3

3 3 3 4 4 5 5 4 4

4 3 3 4 4 5 4 4 3

3 2 3 4 4 5 4 4 4

2 3 3 3 3 4 4 3 3

3 3 4 3 3 4 4 3 4

2 3 3 4 4 3 4 3 3

5 3 3 3 4 4 4 3 5

Mean 3.3 2.9 3.1 3.5 3.6 4.0 3.9 3.4 3.5

**Table 6: Perception evaluation Results**

UA- understandability; RD –Risk documentation; BA – Buildability; UF – usefulness; AC – Accessibility; PA – Portability; SA – Supportability; US – Usability; Ca - Changeability

**Appendices**

**POSTGRADUATE SCHOOL,**

**FEDERAL UNIVERSITY OF AGRICULTURE, ABEOKUTA**

**ONTOLOGY- BASED RISK MANAGEMENT FRAMEWORK FOR SOFTWARE PROJECTS: A SYSTEM LIFECYCLE APPROACH**

Dear Respondent,

The Researcher is from Computer Science Department, Federal University of Agriculture, Abeokuta, Ogun – State, Nigeria. The questionnaire is for a M.Sc. research work and aimed at gathering data on risk management for software projects and pair-wise comparison on all risk groups and risk subclasses in each risk category. It consists of three sections, that is, A – D. you are required to respond to these sections. Kindly respond to the items as honestly as you can, please, be assured that your responses will be treated with utmost confidence and will only be used for academic purposes.

Thank you Sir/Ma.

SECTION A: DEMOGRAPHIC INFORMATION OF RESPONDENTS

1. Organisation Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Department: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Gender: Male[ ] Female[ ]
4. Marital Status: Married[ ] Single[ ] Divorce[ ]
5. Highest Academic Qualification: B.Ed.[ ] B. Sc.[ ] M. Sc.[ ] PhD[ ] others\_\_\_\_\_\_\_\_
6. Job Designation(please specify): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
7. Years of Service: 1 – 10[ ] 11 -20[ ] 21 -30[ ] 31>[ ]

**SECTION B: DISCOVERING RISKS IN SOFTWARE DEVELOPMENT PROJECTS**

1. **REQUIREMENT COLLECTION**

Using previous knowledge of risk may possibly improve future software projects chances of high performance and success. Developing ontology requires the combination of explicit and tacit knowledge. Which of the following method(s) do you use for requirement collection?

* Organisational document
* Literature review
* Expert interview
* Other risk requirement collection or your special methods:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **SOFTWARE RISK IDENTIFICATION**

Which of the following risk identification techniques do you use?

* Questionnaire
* checklists
* review of historical data
* past experience
* expert interview
* analogues
* Using Top-down/ Bottom-up categorization
* Result of experiments or tests
* Review of inadvertent of exposure
* SWOT analysis
* Published materials
* Flowcharts or other business methods
* Others:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **RISK CLASSIFICATION**

* Does the project manager have a method for classifying risks?

1 = Yes, 2 = NO, 3 = Not Sure

If yes, please specify the method(s) used and if No, why?

* Are these methods adequate?

1 = Yes, 2 = NO, 3 = Not Sure

* Have all known management and technical risks been identified?

1 = Yes, 2 = NO, 3 = Not Sure

**SECTION C: PAIR – WISE COMPARISON ON ALL RISK GROUPS AND ALL RISK SUBCLASSES IN EACH RISK CATEGORY**

**Table 1. Scale of relative importance for pairwise comparison.**

|  |  |  |
| --- | --- | --- |
| **Intensity** | **Definition** | **Explanation** |
| **1** | Equal importance | Two activities contribute equally to the object |
| **3** | Moderate importance | Slightly favour one over another |
| **5** | Essential or strong importance | Strongly favour one over another |
| **7** | Demonstrated importance | Dominance of the demonstrated importance in practice |
| **9** | Extreme importance | Evidence favouring one over another of highest possible order of affirmation |
| **2,4,6,8** | Immediate values | When compromise is needed |

**Table 2. Pair-wise comparison on Risk group using information from Table 1.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk Class** | Requirement Analysis | Design | Implementation & unit testing | Integration & System testing | Operation & Maintenance |
| Requirement Analysis |  |  |  |  |  |
| Design |  |  |  |  |  |
| Implementation & unit testing |  |  |  |  |  |
| Integration & System testing |  |  |  |  |  |
| Operation & Maintenance |  |  |  |  |  |

**Table 3a. Pair-wise comparison on Risk Subclass of Requirement Analysis using information from Table1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk Subclass** | Feasibility study | Requirement elicitation | Requirement analysis | Requirement validation | Requirement documentation |
| Feasibility study |  |  |  |  |  |
| Requirement elicitation |  |  |  |  |  |
| Requirement analysis |  |  |  |  |  |
| Requirement validation |  |  |  |  |  |
| Requirement documentation |  |  |  |  |  |

**Table 3b. Pair-wise comparison on Risk Subclass of Design phase using information from Table1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk Subclass** | Document examination | Chosen architectural design | Chosen prog Language | Physical model construction | Design verification | Design specification |
| Document examination |  |  |  |  |  |  |
| Chosen architectural design |  |  |  |  |  |  |
| Chosen prog. Language |  |  |  |  |  |  |
| Physical model construction |  |  |  |  |  |  |
| Design verification |  |  |  |  |  |  |
| Design specification |  |  |  |  |  |  |

**Table 3c. Pair-wise comparison on Risk Subclass of Implementation & unit testing using information from Table1**

|  |  |  |
| --- | --- | --- |
| **Risk Subclass** | Coding | Unit testing |
| Coding |  |  |
| Unit testing |  |  |

**Table 3d. Pair-wise comparison on Risk Subclass of Integration & System testing using information from Table1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk Subclass** | Integration activity | Integration testing | System testing |
| Integration activity |  |  |  |
| Integration testing |  |  |  |
| System testing |  |  |  |

**Table 3e. Pair-wise comparison on Risk Subclass of Operation & Maintenance using information from Table1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Subclass** | Installation | Operation | Acceptance testing | Maintenance |
| Installation |  |  |  |  |
| Operation |  |  |  |  |
| Acceptance testing |  |  |  |  |
| Maintenance |  |  |  |  |

**SECTION D: PAIR – WISE COMPARISON ON ALL RISK FACTORS USING LIKERT’S SCALE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Strongly disagree** | **Disagree** | **Undecided** | **Agree** | **Strongly Agree** |
| **(1)** | **(2)** | **(3)** | **(4)** | **(5)** |

Please tick the one you feel is correct.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S/N** | **RISK FACTORS IN** | **1** | **2** | **3** | **4** | **5** |
|  | **FEASIBILITY STUDY ACTIVITY** |  |  |  |  |  |
| 1. | Under estimation of project time, scope & other resources |  |  |  |  |  |
| 2. | Unrealistic schedule |  |  |  |  |  |
| 3. | Unrealistic budget |  |  |  |  |  |
| 4. | Unclear project scope |  |  |  |  |  |
| 5. | Insufficient resources |  |  |  |  |  |
|  | **REQUIREMENT ELICITATION ACTIVITY** |  |  |  |  |  |
| 6. | Unclear requirement |  |  |  |  |  |
| 7. | Inaccurate requirement |  |  |  |  |  |
| 8. | Conflicting user requirement |  |  |  |  |  |
| 9. | Ignoring the non-functional requirement |  |  |  |  |  |
| 10. | Unclear description of the real environment |  |  |  |  |  |
| 11. | Gold plating |  |  |  |  |  |
|  | **REQUIREMENT ANALYSIS ACTIVITY** |  |  |  |  |  |
| 12. | Infeasible requirement |  |  |  |  |  |
| 13. | Inconsistent requirement |  |  |  |  |  |
| 14. | Non-traceable requirement |  |  |  |  |  |
|  | **REQUIREMENT VALIDATION ACTIVITY** |  |  |  |  |  |
| 15. | Misunderstood domain specific terminology |  |  |  |  |  |
| 16 | Mis-expressing user requirements in natural language |  |  |  |  |  |
|  | **REQUIREMENT DOCUMENTATION ACTIVITY** |  |  |  |  |  |
| 16. | Inconsistent requirement data & requirement document |  |  |  |  |  |
| 17. | Non-modifiable requirement Document |  |  |  |  |  |
|  | **REQUIREMENT DOCUMENT EXAMINATION** |  |  |  |  |  |
| 19. | Requirement Document unclear for developers |  |  |  |  |  |
|  | **CHOSEN ARCHITECTURAL DESIGN METHOD** |  |  |  |  |  |
| 20. | Improper Architectural Design method choice |  |  |  |  |  |
|  | **CHOSEN PROGRAMMING LANGUAGE** |  |  |  |  |  |
| 21. | Improper choice of Programming Language |  |  |  |  |  |
|  | **PHYSICAL MODEL CONSTRUCTION ACTIVITY** |  |  |  |  |  |
| 22. | Complicated design |  |  |  |  |  |
| 23. | Large complex system |  |  |  |  |  |
| 24. | Unavailable expertise for reusability |  |  |  |  |  |
|  | **DESIGN VERIFICATION ACTIVITY** |  |  |  |  |  |
| 25. | Difficulties in verifying design to requirement |  |  |  |  |  |
| 26. | Many feasible solution |  |  |  |  |  |
| 27. | Wrong design |  |  |  |  |  |
|  | **DESIGN SPECIFICATION ACTIVITY** |  |  |  |  |  |
| 28. | Difficulties in allocating functions to components |  |  |  |  |  |
| 29. | Extensive specification |  |  |  |  |  |
| 30. | Omitting data processing functions |  |  |  |  |  |
|  | **DESIGN ACTIVITY DOCUMENTATION** |  |  |  |  |  |
| 31. | Incomplete design document |  |  |  |  |  |
| 32. | Large design document |  |  |  |  |  |
| 33. | Unclear design document |  |  |  |  |  |
| 34. | Inconsistent design document |  |  |  |  |  |
|  | **CODING ACTIVITY** |  |  |  |  |  |
| 35. | Non-readable design document |  |  |  |  |  |
| 36. | Developing the wrong user interface |  |  |  |  |  |
| 37. | Programming language does not support architectural design |  |  |  |  |  |
| 38. | Modules are developed by different programmers |  |  |  |  |  |
| 39. | Complex, ambiguous, inconsistent code |  |  |  |  |  |
| 40. | Large amount of repetitive code |  |  |  |  |  |
| 41. | Inexperienced programmers |  |  |  |  |  |
| 42. | Too many syntax error |  |  |  |  |  |
| 43. | Technology change |  |  |  |  |  |
|  | **UNIT TESTING ACTIVITY** |  |  |  |  |  |
| 44. | High fault rate in newly design components |  |  |  |  |  |
| 45. | Code is not understandable by reviewers |  |  |  |  |  |
| 46. | Lack of complete automated tools |  |  |  |  |  |
| 47. | Not all faults are discovered in unit testing |  |  |  |  |  |
| 48. | Testing is monotonous, boring & repetitive |  |  |  |  |  |
| 49. | Coding Drivers and stubs |  |  |  |  |  |
| 50. | Poor regression testing |  |  |  |  |  |
|  | **INTEGRATION ACTIVITY** |  |  |  |  |  |
| 51. | Difficulties in ordering components’ integration |  |  |  |  |  |
| 52. | Integrate wrong version of components |  |  |  |  |  |
| 53. | Omissions or oversights |  |  |  |  |  |
|  | **INTEGRATION TESTING ACTIVITY** |  |  |  |  |  |
| 54. | A lot of bugs emerged during the integration |  |  |  |  |  |
| 55. | Data loss across an interface |  |  |  |  |  |
| 56. | Difficulties in localizing errors |  |  |  |  |  |
| 57. | Difficulties in repairing errors |  |  |  |  |  |
|  | **SYSTEM TESTING ACTIVITY** |  |  |  |  |  |
| 58. | Unqualified testing team |  |  |  |  |  |
| 59. | Limited testing resources |  |  |  |  |  |
| 60. | Inability to test in the operational environment |  |  |  |  |  |
| 61. | Testers rely on process myths |  |  |  |  |  |
| 62. | Testing cannot cope with requirement change |  |  |  |  |  |
| 63. | Non-testable system |  |  |  |  |  |
|  | **INSTALLATION ACTIVITY** |  |  |  |  |  |
| 64. | Problems in installation |  |  |  |  |  |
| 65. | The effect on the environment |  |  |  |  |  |
| 66. | Environmental change |  |  |  |  |  |
|  | **OPERATION ACTIVITY** |  |  |  |  |  |
| 67. | Emergence of new requirement |  |  |  |  |  |
| 68. | Difficulties in using the system |  |  |  |  |  |
|  | **ACCEPTANCE TESTING ACTIVITY** |  |  |  |  |  |
| 69. | User resistance to change |  |  |  |  |  |
| 70. | Missing capabilities |  |  |  |  |  |
| 71. | Too many software faults |  |  |  |  |  |
| 72. | Testers under performance |  |  |  |  |  |
| 73. | Suspension & resumption problems |  |  |  |  |  |
| 74. | Insufficient data handling |  |  |  |  |  |
|  | **MAINTENANCE ACTIVITY** |  |  |  |  |  |
| 75. | The software Engineer cannot reproduce the problem |  |  |  |  |  |
| 76. | Problems in maintainability |  |  |  |  |  |
| 77. | Budget contention |  |  |  |  |  |

**APPENDIX B – PERFORMANCE APPRAISAL FORM (PAF)**

**DEPARTMENT OF COMPUTER SCIENCE,**

**COLLEGE OF PYSICAL SCIENCES**

**POSTGRADUATE SCHOOL,**

**FEDERAL UNIVERSITY OF AGRICULTURE, ABEOKUTA**

**PROJECT TOPIC:** Ontology-based Risk Management Framework for Software project: A software Lifecycle Approach

**PERFORMANCE APPRAISAL FORM (PAF)**

Dear respondent,

This appraisal form is being administered to you in respect of a research being conducted to find out the performance of the framework and how useful it is in relation to software development. Kindly complete this form with absolute honesty bearing in mind that your responses are strictly for research purpose.

**Section A: Expert‘s Details**

Name:………………………………………………………………………………………………………………………………………………….

Age:…………………………………………………………………………………………………………………………………………………....

Years of Experience:…………………………………………………………………………………………………………………………….

Job Description:…………………………………………………………………………………………………………………………………..

Sex: Male ( ) Female ( )………………………………………………………………………………………………………………………..

Educational Qualification:……………………………………………………………………………………………………………………

**Section B: Successful Software Project Indicators Assessment**

Please tick the column that best suit your use. The response models for this section are: Exceptional (5), Exceeds Expectation (4), Meet Expectation (3), Below Expectation (2) and Needs Expectation (1).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Indicators** | **Exceptional**  **(5)** | **Exceeds Expectation**  **(4)** | **Meet Expectation**  **(3)** | **Below Expectation**  **(2)** | **Needs Expectation**  **(1)** |
| Cost |  |  |  |  |  |
| Duration(time) |  |  |  |  |  |
| Complexity |  |  |  |  |  |
| Quantity |  |  |  |  |  |
| Scope |  |  |  |  |  |
| Functional Requirement |  |  |  |  |  |

**APPENDIX C – PERCEPTION FORM**

**DEPARTMENT OF COMPUTER SCIENCE**

**COLLEGE OF PYSICAL SCIENCES**

**POSTGRADUATE SCHOOL,**

**FEDERAL UNIVERSITY OF AGRICULTURE, ABEOKUTA**

**PROJECT TOPIC:** Ontology-based Risk Management Framework for Software project: A software Lifecycle Approach

**Software Experts’ Perception on the Usage of Risk Management Framework for Software Project**

Dear respondent,

This perception form is being administered to you in respect of a research being conducted to find out the perception of the framework and how useful it is in relation to software development. Kindly complete this form with absolute honesty bearing in mind that your responses are strictly for research purpose.

**Section A: Expert‘s Details**

Name:………………………………………………………………………………………………………………………………………………….

Age:…………………………………………………………………………………………………………………………………………………....

Years of Experience:…………………………………………………………………………………………………………………………….

Job Description:…………………………………………………………………………………………………………………………………..

Sex: Male ( ) Female ( )………………………………………………………………………………………………………………………..

Educational Qualification:……………………………………………………………………………………………………………………

**SECTION B: Personal Attribute**

**INSTRUCTION**

Please tick the column that best suit your use. The response models for this section are: Exceptional (5), Exceeds Expectation (4), Meet Expectation (3), Below Expectation (2) and Needs Expectation (1).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Exceptional**  **(5)** | **Exceeds expectation**  **(4)** | **Meet expectation**  **(3)** | **Below expectation**  **(2)** | **Needs expectation**  **(1)** |
| **Understandability** -How straight forward is it to understand |  |  |  |  |  |
| **Risk documentation** – Does it actually capture the likeable risks that combat successful completion of software risks |  |  |  |  |  |
| **Buildability** – How straight forward has this framework helps in meeting pre- requisite for developing a successful software project? |  |  |  |  |  |
| **Usefulness** – To what extent has this framework being useful within its application domain? |  |  |  |  |  |
| **Accessibility** – To what extent is this framework accessible (binary distribution available)? |  |  |  |  |  |
| **Portability** – To what extent can the software be used on other platforms? |  |  |  |  |  |
| **Supportability** – To what extent has this framework support or contributes to the success of your work? |  |  |  |  |  |
| **Usability** – is there any difficulty in the usable of this framework |  |  |  |  |  |
| **Changeability** – How easy is it to modify this framework in order to add newly discovered risk index? |  |  |  |  |  |