IJCSIS Vol. 13 No. 7, July 2015
ISSN 1947-5500

International Journal of Computer Science & Information Security

Cornell University Library

Cogprints

Google scholar

.docstoc
find and share professional documents

ScientificCommons

View my documents on Scribd

BASE
Bielefeld Academic Search Engine

scirus
search engine for science

SciRate.com

CiteSeer[x] beta

dblp.uni-trier.de
Computer Science Bibliography

Q·Sensei BETA

DOAJ DIRECTORY OF OPEN ACCESS JOURNALS

EBSCO HOST

ProQuest

IJCSIS

ISSN (online): 1947-5500

Please consider to contribute to and/or forward to the appropriate groups the following opportunity to submit and publish original scientific results.

CALL FOR PAPERS
International Journal of Computer Science and Information Security (IJCSIS)
January-December 2015 Issues

The topics suggested by this issue can be discussed in term of concepts, surveys, state of the art, research, standards, implementations, running experiments, applications, and industrial case studies. Authors are invited to submit complete unpublished papers, which are not under review in any other conference or journal in the following, but not limited to, topic areas.
See authors guide for manuscript preparation and submission guidelines.

Indexed by Google Scholar, DBLP, CiteSeerX, Directory for Open Access Journal (DOAJ), Bielefeld Academic Search Engine (BASE), SCIRUS, Scopus Database, Cornell University Library, ScientificCommons, ProQuest, EBSCO and more.

Deadline: **see web site**
Notification: see web site
Revision: see web site
Publication: see web site

Context-aware systems
Networking technologies
Security in network, systems, and applications
Evolutionary computation
Industrial systems
Evolutionary computation
Autonomic and autonomous systems
Bio-technologies
Knowledge data systems
Mobile and distance education
Intelligent techniques, logics and systems
Knowledge processing
Information technologies
Internet and web technologies
Digital information processing
Cognitive science and knowledge
Agent-based systems
Mobility and multimedia systems
Systems performance
Networking and telecommunications
Software development and deployment
Knowledge virtualization
Systems and networks on the chip
Knowledge for global defense
Information Systems [IS]
IPv6 Today - Technology and deployment
Modeling
Software Engineering
Optimization
Complexity
Natural Language Processing
Speech Synthesis
Data Mining

For more topics, please see web site https://sites.google.com/site/ijcsis/

For more information, please visit the journal website (https://sites.google.com/site/ijcsis/)

Editorial
Message from Managing Editor

The International Journal of Computer Science and Information Security (IJCSIS) is a comprehensive and refereed, international publication featuring emerging research findings and industry solutions involving all aspects of computing and security. The editorial board is pleased to present the July 2015 issue. The purpose of this edition seeks to improve and expand existing research in experimental and theoretical research from both industry and academia in the broad areas of Computer Science, ICT & Security and related areas. As the editors of this issue, we are glad to see variety of articles focusing on the major topics of innovation and computer science; high performance computing; security; genetic algorithms; interdisciplinary applications & mobile technologies etc. This scholarly resource endeavors to provide international audiences with the highest quality research manuscripts and accounts of the constant evolution of information science and technology in whole. Researchers, academicians, practitioners and students will find this journal as a critical source of reference for all advanced technological applications and developments.

Over the last years, we have witnessed significant growth of IJCSIS in several key areas, include the expansion of scope to recruit papers from emerging areas of green & sustainable computing, cloud computing security, forensics, mobile computing and big data analytics. IJCSIS archives all publications in major academic/scientific databases and is indexed by the following International agencies and institutions: Google Scholar, CiteSeerX, Cornell's University Library, Ei Compendex, Scopus, DBLP, DOAJ, ProQuest, ArXiv, ResearchGate and EBSCO.

We thank and congratulate the wonderful team of publication staff members, associate editors, and reviewers for their dedicated services to select and publish extremely high quality papers for publication in IJCSIS. In particular, we would like to thank all authors for submitting their high quality papers to IJCSIS and the readers for continued support to IJCSIS by citing papers published in IJCSIS. Without their continued and unselfish commitments, IJCSIS would not have achieved its current premier status.

"We support researchers to succeed by providing high visibility & impact value, prestige and excellence in research publication."

*For further questions please do not hesitate to contact us at **ijcsiseditor@gmail.com**.*

A complete list of journals can be found at:
http://sites.google.com/site/ijcsis/

IJCSIS Vol. 13, No. 7, July 2015 Edition

ISSN 1947-5500 © IJCSIS, USA.

Journal Indexed by (among others):

Bibliographic Information
ISSN: 1947-5500
Monthly publication (Regular Special Issues)
Commenced Publication since May 2009

Editorial / Paper Submissions:
IJCSIS Managing Editor
(ijcsiseditor@gmail.com)
Pennsylvania, USA
Tel: +1 412 390 5159

IJCSIS EDITORIAL BOARD

TABLE OF CONTENTS

Index Terms—Cloud Computing Security, Infrastructure-as-a-Service(IAAS), Platform-as-a-Service(PAAS), Software-as-a-Service(SAAS), Private Cloud, Public Cloud, Hybrid Cloud, Trust, Vulnerabilities.

4. Paper 30061514: A Framework for Web Search Log Evaluation for Testing Information Search in Cloud (pp. 29-33)

Althaf Ali A, Bharathiar University, Coimbatore, Tamilnadu, India
Dr. R. Mahammad Shafi, Dept. Of MCA, Sree Vidyaniketan Engineering College, Tirupati, India.

Abstract — To meet the rapid growth of cloud technologies, many web information provider application are developed and deployed, and these applications run in the cloud. Because of the scalability provided by the clouds, a Web application can be visiting by several millions or billions of users. Therefore, the testing and evaluation of performance of these applications are becoming increasingly important. Web application usage log evaluation is one of the promising approaches to tackle the performance problem by adapting the content and structure of application to the needs of the users by taking advantage of the knowledge acquired from the analysis of the users searching activities from the web search logs. We propose a framework for web search log evaluation using classification and clustering method for effective testing information search in cloud. It also provides an information search ranking method to refine and optimizes the search evaluation process. We evaluate the proposed approach through implementing a web proxy in a server to record the user search logs and measure the retrieval precision rate for different users. A rate of 25% precision improvement is observed using different cluster testing for different users.

Keywords- Cloud, Web Search, Web Log, Classification, Clustering, Information Search, Testing.

5. Paper 30061527: An Integrated Mobile Application for Enhancing Management of Nutrition Information: Case Study of Arusha Region (pp. 34-38)

Neema Mduma, Khamisi Kalegele, School of Computation and Communication Science &Engineering, Nelson Mandela African Institution of Science and Technology, Arusha, Tanzania

Abstract — Based on the fact that management of nutrition information is still a problem in many developing countries including Tanzania and nutrition information is only verbally provided without emphasis, this study proposes mobile application for enhancing management of nutrition information. The paper discusses the implementation of an integrated mobile application for enhancing management of nutrition information based on literature review and interviews, which were conducted in Arusha region for the collection of key information and details required for designing the mobile application. In this application, PHP technique has been used to build the application logic and MySQL technology for developing the back-end database. Using XML and Java, we have built an application interface that provides easy interactive view.

Keywords - Nutrition information; MySQL; XML; Java; PHP; Mobile Application.

6. Paper 30061533: Automated Vehicle Tracking by GPS Modern Technology (pp. 39-48)

Fahmi Ahmed, Faculty of Engineering, University Of Development Alternative (UODA), Dhaka, Bangladesh
A.B.M. Rocknuzzaman, Faculty of Engineering, Universität Duisburg-Essen, Duisburg, Germany
Syed Foysol Islam, Faculty of Engineering, University Of Development Alternative (UODA), Dhaka, Bangladesh

Abstract — This Research paper represents, Automatic vehicle tracking by GPS modern system. The results obtained in a purpose-designed computer software for track Vehicle's location and other data using by modern GPS or GLONASS technology. For this system need two types of product, one is technology product and another is web system.

Keywords - Microcontroller, IC Max232, GPS Antenna, GPRS and GSM Antenna, Satellite Communication, UART Communication, GPS, GPRS, GSM, Web Application, Google API, Open Street AP.

7. Paper 30061543: Using J48 Tree for Value-Based Customer Relations Management (CRM) (pp. 49-55)

Marzieh mohammadi, Department of Computer Engineering, Najafabad branch, Islamic Azad University, Isfahan, Iran
Hamid Rastegari, Department of Computer Engineering, Najafabad branch, Islamic Azad University, Isfahan, Iran

Abstract — One of the main important issues in critical to retail success is decision support methods for marketing decisions. Different data mining techniques can be suitable for targeted marketing and efficient customer segmentation. Mainly over data mining, the extraction of hidden predictive pattern from datasets organizations can recognize forecast future behaviors profitable customers, and assist firms to create proactive, knowledge-driven choices. The mechanized, future-oriented analyses is possible with data mining move outside the analyses of previous events usually provided with history-oriented tools like decision support systems. Data mining techniques response business requests that in the previous were too time consuming to follow. However, the responses to these requests create customer relationship management probable. Therefore, in this paper, a model base on the classification of J48 tree and feature selection is proposed to predict precise marketing performance. The propose model is evaluated conducted 3datasets and the results are compared with other algorithms such as Rep tree, Random tree and J48 tree. The experimental results show that the proposed model has higher precision and lower error rate in comparison of J48 tree, Rep tree and Random tree.

Keywords-Customer relations management (CRM); Feature Selection; Data mining; Classification; J48 tree

8. Paper 30111406: A Security Architecture for Software Defined Networks (SDN) (pp. 56-61)

Okunade Oluwasogo Adekunle, School of Science and Technology, National Open University of Nigeria, Victoria Island, Lagos, Nigeria.
Osunade Oluwaseyitan, Department of Computer Science, University of Ibadan, Ibadan, Nigeria.

Abstract - Software defined networking is an emerging network architecture with promising future in network field. It is dynamic, manageable, cost effective, and adaptable networking where control and data plane are decoupled, and control plane is centrally located to control application and dataplanes. OpenFlow is an example of Software Defined Networking (SDN) Southbound, which provides an open standard based interface between the SDN controller and data plane to control how data packets are forwarded through the network. As a result of rapid changes in networking, network program-ability and control logic centralization capabilities introduces new fault and easily attack planes, that open doors for threats that did not exist before or harder to exploit. This paper proposed SDN architecture with some level of security control, this will provide secured SDN paradigm with machine learning white/black list, where users application can be easily test and group as malicious attack or legitimate packet.

Keyword - Software Defined Networking (SDN); OpenFow; Flow table; Security control; white/black list

9. Paper 30061502: Design and Extraction of Facial Components in Automated Face Recognition (pp. 62-65)

Ms. Roshani R. Patil, Mr. P. S. Mohod, Department of Computer Science and Engineering G. H. Raisoni College of Engineering Nagpur, Maharashtra, India

Abstract - Face recognition presents a challenging problem in the field of image analysis and computer vision. Face recognition system should be able to automatically detect a face in an image. This involves extracts its features and then recognize it, regardless of lighting, expression, illumination, ageing, transformations (translate, rotate and scale image) and pose, which is a difficult task. This paper presents a framework for component- based face alignment and representation that demonstrates improvement in matching performance over the more common holistic approach to face alignment and representation. Active shape model (ASM) technique that has been used often for locating facial features in face images. The proposed scheme selects robust landmark points where relevant facial features are found and assigns higher weights to their corresponding features in the face classification stage. For alignment and cropping Procrutes analysis is used. Multi-scale local binary pattern is used for matching automated face image. In MLBP per-component measurement of facial similarity and fusion of per-component similarities is used. The proposed work is more robust to changes in facial pose and improves recognition accuracy on occluded face images in forensic scenarios.

Keywords:- Active shape model, Multi-scale local binary pattern, Procrutes analysis, holistic method.

10. Paper 30061503: Software Reliability Estimation of Component Based Software System using Fuzzy Logic (pp. 66-71)

Gopal Prasad Jaiswal, Ram Nivas Giri, Department of Computer Science and Engineering, RITEE Raipur, Chhattisgarh, CSVTU Bhilai, Chhattisgarh, India

Abstract — Software Reliability Modeling has been one of the much-attracted research domains in Software Reliability Engineering. Software reliability means provide reusable, less complex software, to perform a set of successful operation and his function within a provided time and environment. Software designers are motivated to develop reliable, reusable and useful software. In past, Object-Oriented Programming System (OOPS) concept is to be used in purpose of reusability but they are not providing powerful to cope with the successive changing as per requirements of ongoing applications. After that Component Based Software system (CBSS) is in floor. IT is based on reusability of his component with less complexity. This paper presents a new approach to analyze the reusability, dependency, and operation profile as well as application complexity of component-based software system. Here, we apply Fuzzy Logic approach to estimate the reliability of component-based software system with the basis of reliability factor.

Index Terms—Component, Object-Oriented Programming System (OOPS), Component Based Software system (CBSS), Fuzzy Logic, Fuzzy Inference System (FIS), Adaptive Neuro Fuzzy Inference System (ANFIS), Reliability, Application Complexity, Component Dependency, Operation Profile, Reusability, Fuzzification, Defuzzification, Reliability Model, Rule Based Model, Path Based Model, Additive Model, etc.

11. Paper 30061536: Constructing an Add-in Tool for Enterprise Architect v7.5 To Measure the Quality of Object Oriented Design (Class Diagram) (pp. 72-85)

Laheeb Mohammed Ibrahim, Khalil Ahmed Ibrahim, Software Engineering, Mosul University, Collage of Computer Sc. & Math., Mosul, Iraq

Abstract — Software design is very important stage in software engineering since it lies in the middle of the software development life cycle and costs can be reduced if corrections or improvements made in design phase. Some of the existing CASE tools do not have the ability to correct or improve software design like EA v7.5. The present study aims to construct a CASE tool that helps software engineers in design phase by assessing or evaluating the quality of that design using object oriented design metrics, use the developed CASE tool as add-in to work inside Enterprise Architect since it has no support for design metrics. So, this paper may be considered as an evolvement of such a well-known CASE tool like the Enterprise Architect. In this paper, three tools are developed. First, is "K Design Metrics tool (KDM)" as an add-in that works inside Enterprise Architect (EA) v7.5 which is a well-known, powerful CASE (Computer Aided Software Engineering) tool. KDM tool takes the XMI (XML Metadata Interchange) document for the UML class diagram exported by EA as input, processes it, calculates and visualize metrics, provides recommendations about design naming conventions and exports metrics as XML (Extensible Markup Language) document in order to communicate with other tools namely KRS (K Reporting Service) and KDB (K Database). A Second tool is K Reporting Service (KRS) "KRS" which takes XML document generated by KDM tool as input, parses it and gives a report. The report helps the project manager or the team leader to monitor the progress and to document the metrics. Hence KRS tool is integrated with Enterprise Architect. Lastly, K Database "KDB" which takes the same XML document generated by KDM tool as input, parses it and stores metrics in the database to be used as a historical data. KDB tool is also integrated with Enterprise Architect. Two object oriented design metrics models are used, namely MOOD (Metrics for Object Oriented Design) which measures

Encapsulation, Inheritance, Polymorphism and Coupling, and MEMOOD (Maintainability Estimation Model for Object Oriented software in Design phase) which measures Understandability, Modifiability and Maintainability. Both models are validated theoretically and empirically. These measurements allow designers to access the software early in process, make changes that will reduce complexity and improve the design. All three tools were developed using C# programming language with the aid of Microsoft Visual Studio 2010 as integrated development environment under Windows 7 operating system with minimum 4 GB of RAM and Core-i3 of CPU.

Keywords-MOOD (Metrics for Object Oriented Design); *MEMOOD (Maintainability Estimation Model for Object Oriented software in Design phase)*; *UML (Unified Modeling Language)*; *Object Oriented software*; *Enterprise Architect v7.5.*

12. Paper 30061517: Usability Evaluation Methods and Principles for the Web (pp. 86-92)

Joel Mvungi, Titus Tossy, Computer Science Studies Department, Mzumbe University, Morogoro, Tanzania

Abstract - In order to determine the quality of any web application in the world, Usability is the one of the most important tool that one can use. Web analysis perform several inspections on the websites and software and use usability criteria to determine some faults on the systems. Usability engineering has being important tool for the companies as well, this is due to the fact that through usability engineering companies can improve their market level by making their products and services more accessible. Know days there some web application and software products which are complex and very sophisticated, hence usability can be able to determine their success or failure. However currently usability has been among the important goal for the Web engineering research and much attention is given to usability by the industry due to recognition of the importance of adopting usability evolution methods before and after deployment. Moreover several literature has proposed several techniques and methods for evaluating web usability. And however there is no agreement yet in the software on which usability evolution method is better than the other. Extensive usability evaluation is usually not feasible for the case of web development process. In other words unusable website increases the total cost of ownership, and therefore this paper introduces principles and evaluation methods to be used during the whole application lifecycle, so as to enhance usability of web applications.

Keywords - Evolution methods, Web usability, Web usability principles, Development process.

Constructing an Add-in Tool for Enterprise Architect v7.5 To Measure the Quality of Object Oriented Design (Class Diagram)

Laheeb Mohammed Ibrahim
Software Engineering
Mosul University, Collage of Computer Sc. & Math.
Mosul , Iraq

Khalil Ahmed Ibrahim
Software Engineering
Mosul University, Collage of Computer Sc. & Math.
Mosul , Iraq

***Abstract*— Software design is very important stage in software engineering since it lies in the middle of the software development life cycle and costs can be reduced if corrections or improvements made in design phase. Some of the existing CASE tools do not have the ability to correct or improve software design like EA v7.5.**

The present study aims to construct a CASE tool that helps software engineers in design phase by assessing or evaluating the quality of that design using object oriented design metrics, use the developed CASE tool as add-in to work inside Enterprise Architect since it has no support for design metrics. So, this paper may be considered as an evolvement of such a well-known CASE tool like the Enterprise Architect

In this paper, three tools are developed. First, is "K Design Metrics tool (KDM)" as an add-in that works inside Enterprise Architect (EA) v7.5 which is a well-known, powerful CASE (Computer Aided Software Engineering) tool. KDM tool takes the XMI (XML Metadata Interchange) document for the UML class diagram exported by EA as input, processes it, calculates and visualize metrics, provides recommendations about design naming conventions and exports metrics as XML (Extensible Markup Language) document in order to communicate with other tools namely KRS (K Reporting Service) and KDB (K Database).

A Second tool is K Reporting Service (KRS) "KRS" which takes XML document generated by KDM tool as input, parses it and gives a report. The report helps the project manager or the team leader to monitor the progress and to document the metrics. Hence KRS tool is integrated with Enterprise Architect.

Lastly, K Database "KDB" which takes the same XML document generated by KDM tool as input, parses it and stores metrics in the database to be used as a historical data. KDB tool is also integrated with Enterprise Architect.

Two object oriented design metrics models are used, namely MOOD (Metrics for Object Oriented Design) which measures Encapsulation, Inheritance, Polymorphism and Coupling, and MEMOOD (Maintainability Estimation Model for Object Oriented software in Design phase) which measures Understandability, Modifiability and Maintainability. Both models are validated theoretically and empirically. These measurements allow designers to access the software early in process, make changes that will reduce complexity and improve the design.

All three tools were developed using C# programming language with the aid of Microsoft Visual Studio 2010 as integrated development environment under Windows 7 operating system with minimum 4 GB of RAM and Core-i3 of CPU.

***Keywords-MOOD (Metrics for Object Oriented Design)*; *MEMOOD (Maintainability Estimation Model for Object Oriented software in Design phase)*; *UML (Unified Modeling Language)*; *Object Oriented software*; *Enterprise Architect v7.5*.**

I. Introduction

Software design (object oriented) is the stage in the software engineering process where the executable software system is developed. So, it plays a pivotal role in software development since it determines the structure of the software solution. Once the design has been implemented, it is difficult and expensive to change. Therefore, high design quality is vital for reducing software cost [23] [3] [17][34]

Quality assurance plays an important role in monitoring software process in the form of umbrella activities (Umbrella activities are applied throughout a software project and help a software team manage and control progress, quality, change, and risk [15]) and in the form of measurement or metrics. Without measurements (or metrics), it is impossible to detect problems early in the software process, before they get out of hand. Metrics therefore can evaluate the process and serve as an early warning system for potential problems [20].

Many object oriented design metrics have been proposed specifically for the purpose of assessing the design of a software system such as MOOD (Metrics for Object Oriented Design), CK (Chidamber and Kemrer), Lorenz and Kids metrics [12]. Some of these metrics (or models) are supported by CASE tools due to their importance in evaluating or assessing the design of the software system.

Enterprise Architect (EA) is a well-known CASE tool that is used in over 130 countries for designing and constructing software systems. EA differentiates from other tools in that it supports a comprehensive UML modeling, have a built-in requirements management, test management, extensive project management support, Code engineering, and .. Etc. But it does not support metrics on software design [25].

As been mentioned earlier that software design is a very important stage in software engineering since it lies in the middle of the software development life cycle and costs can be reduced if corrections or improvements made in design phase. Some of the existing CASE tools do not have the ability to correct or improve software design like EA v7.5.

The present paper aims to construct a CASE tool that helps software engineers in design phase by assessing or evaluating the quality of that design using object oriented design metrics, using two metrics models namely MOOD (Metrics for Object Oriented Design) which measures Encapsulation, Inheritance, Polymorphism and Coupling, and MEMOOD (Maintainability Estimation Model for Object Oriented Systems in Design phase) which measures understandability, modifiability and maintainability, and using the developed CASE tool as add-in to work inside EA since it has no support for design metrics. So, this paper may be considered as an evolvement of such a well-known CASE tool like the EA v7.5

II. RELATED WORK

Many researchers have worked on object oriented design by means of quality assurance. Some of them propose tools that calculate metrics, other have made surveys about quality models. Following a brief explanation about their works:

Paterson, T et al. (2002) demonstrated the potential for deriving a suite of object-oriented design metrics by the XSLT (Extensible Style Sheet Language Transformation) processing of XMI representations of UML class diagram models. They propose a tool that extracts metrics like number of classes [13].

Girgis, M.R et al. (2009) proposed a tool that automates the computation of the important metrics that are applicable to the UML class diagrams. The tool collects information by parsing the XMI format of the class diagram, and then uses the data to calculate the metrics like CK, MOOD [6].

Poornima, U.S (2011) stated that quality metrics are helpful for the designers in measuring solution architecture for better products. By understanding the solution domain of object oriented systems and measuring the quality of the design using metrics yields to future enhancements [14].

Mago, J. et al. (2012) proposed a model based on fuzzy logic which serves as an integrated means to provide an interpretation of the object oriented design metrics and also surveyed MOOD metrics with other metrics [11].

Rani, T. et al. (2012) proposed a tool (SD-Metrics) that measures the complexity of a class diagram using class metrics from XMI files from Argo UML [16].

Sharma, A.K. et al. (2012) reviewed quality metrics suites namely, MOOD, CK and Lorenz & Kidd, selected some metrics and discarded others based on the definition and capability of the metrics [22].

Hilera, J.R. et al. (2012) made a web service for calculating the metrics of UML class diagrams from XMI document. They stated that as UML becomes a standard format for specifying classes, it is useful to have a web service that quickly runs metrics on the diagram and gives developers feedback on the class quality [7].

Jassim F. et al. (2013), the main goal was to predict factors of MOOD metrics for object oriented design using a statistical approach. They also used a linear regression model to find out the relationship between factors of MOOD and their influence on object oriented software measurements [10].

Ahmed S.H. et al. (2013), proposed a hybrid metrics suite for evaluating the design of object oriented software early in UML design phase. A metrics extraction tool was developed which operated on UML design models and corresponded XMI files to assure independency results [2].

All studies state that design metrics are important to access the software design early in process and make changes that will improve the design. None of the above mentioned studies fully automate MOOD metrics. In this paper, all MOOD metrics were fully automated and another model (MEMOOD) is used as an add-in inside EA. None of the above studies integrates or improves an existing CASE tool.

III. SOFTWARE ENGINEERING AND QUALITY ASSURANCE

According to [9], ***Software Engineering*** can be defined as the "application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software". Building an information system using the Software Development Life Cycle (SDLC) follows a similar set of phases see Fig. 1, requirements phase, design phase, implementation phase, test phase, installation/checkout phase, and operation/maintenance phase [22][26][35].

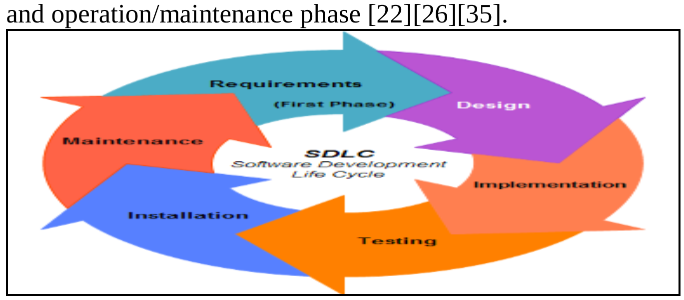

Figure 1. Software Development Life Cycle (SDLC)

Quality must be defined and measured if improvement is to be achieved. Yet, a major problem in quality engineering and management is that the term 'Quality' is ambiguous, so it is commonly misunderstood. The confusion may be attributed to several reasons. First, quality is not a single idea, but rather a multidimensional concept. Second, for any concept there are levels of abstraction; when people talk about quality, one party could be referring to it in its wide sense, whereas another might be referring to its specific meaning. Third, the term quality is a part of the daily language; the popular and professional uses of it may be very different [32]. So, just to be clear, ***Quality*** can be defined as [9]. The degree to which a system, component, or process meets a customer or user's needs or expectations.

A key element of any engineering process is measurement. Using measures allows for better understanding of the

http://sites.google.com/site/ijcsis/
ISSN 1947-5500

attributes of the models that will be created and assessing the quality of the engineered products or systems to be built.

Measure is defined as a quantitative indication of the extent, amount, dimension, capacity, or size of some attribute of a product or process whereas ***Measurement*** is the act of determining a measure [15]. ***Metric*** is "a quantitative measure of the degree to which a system, component, or process possesses a given attribute [9].When a single data point has been collected (e.g., the number of errors uncovered within a single software component), a measure has been established. Measurement occurs as the result of the collection of one or more data points. Software metric relates the individual measures in some way (e.g., the average number of errors found per review) [15].

IV. ENTERPTISE ARCHITECT (EA)

EA is a CASE tool for designing and constructing software systems, for business process modeling, and for more generalized modeling purposes [30][28]. EA was developed by Sparx Systems © and it covers all aspects of the software development cycle from requirements gathering, through analysis, model design, testing, change control and maintenance to implementation, with full traceability (identifies the way a given process has been, or is to be, developed in a system). [30].

EA has proven to be highly popular across a wide range of industries and is used by thousands of companies worldwide, from large, well known, multinational organizations to smaller independent companies and consultants [24]. Sparx Systems © software is used in the development of many kinds of applications and systems in a wide range of industries, including aerospace, banking, web development, engineering, finance, medicine, military, research, academia, transport, retail, utilities (such as gas and electricity) and electrical engineering. It is also used effectively for UML and enterprise architecture training in many prominent colleges, training companies and universities around the world [24] [29]. For all those reasons mentioned earlier in addition to the powerful description of UML class diagrams as XMI, this paper tends to use EA as a platform for the proposed tools to work with.

EA is a great UML CASE tool, but we can make it even better by adding and extending new functionality in the form of an add-in**.** To fully understand the steps necessary to get the add-in running, we should first understand how EA's add-in architecture works[33]. When EA starts up, it will read the registry key [HKEY _ CURRENT _ USER \ Software \ Sparx Systems \EAAddins].

Each of the keys in this location represents an add-in for EA to load. The (default) value of the key contains the name of the assembly and the name of the add-in class separated by a dot. EA then asks Windows for the location of the assembly (An assembly is a file that is automatically generated by the .NET compiler upon successful compilation of every .NET application. It can be either a DLL or an executable file), which is stored on the COM codebase entries in the registry, and it will use the public operations defined in the add-in class[33].

V. ANALYSIS AND DESIGN OF KDM, KRS AND KDB TOOLS

This section explains in detail the proposed tools from the analysis and design point of view. These tools are named KDM, KRS and KDB Tool, which helps the software engineer in the design phase of the software life cycle. For modeling the proposed tools, The following CASE tools (Edraw Max, Microsoft Visio and EA) are used.

Before start analyzing the proposed tools in detail, it is needed to describes them in a general way by showing how the final user of the proposed tools like a software engineer, project manager or programmer will use them. KDM tool is used to calculate the metrics for the OOD and considered being the main tool, while KRS tool can help with the documentation of the results, and finally KDB tool can help by storing the metrics in the database.

A. K-Design Metrics (KDM) Tool

EA does not support any tool that measures the class diagram. So, in this paper KDM tool was developed to work from inside the EA as add-in to help software engineer understanding the design of the software better by scrutinizing the class diagram of that software by means of design metrics.
In addition, the KDM tool (add-in) can be deployed to work on other machines not just on the machine where it is developed, so that other software engineers can use it. The proposed KDM tool accepts XMI 1.1 for the UML 1.3 generated by the EA v7.5 as **input** and calculates metrics for that design.
The **output** of KDM tool is the value of metrics and 3-dimension pie chart which visualizes the value of each metric. It also gives statistics about that design and produces XML document. See Fig 2 which shows the input and output of the KDM tool.

1. KDM Tool in SDLC

KDM tool operates on UML class diagram either in the analysis phase (high-level design) or in the design phase (low-level or detailed design). KDM tool is classified as Upper CASE Tool (front-end) since it works in the upper level of the SDLC.

2. How KDM Tool Works

KDM tool imports XMI document which then will be fed into the XMI parser. The parser will extract the required information from XMI document and pass it to the metric module which contains the MOOD model and MEMOOD model which in turn calculates the metrics for that design. KDM tool draws 3D pie chart, gives recommendations about design naming conventions, and gives design statistics, also exports XML document.

XMI is a way of saving UML diagrams as XML so it contains huge data that describes the UML diagram (in this paper the class diagram) in detail such as the name of each class, its attributes, operations, relationships, style, etc.

XMI document is stored either as XML or XMI extension which means that the information is represented or structured as **tags**.

XMI document has a large set of tags. Some are important but others are not, such as the style of each class, date of creation, etc. The tags used to calculate the metrics in this paper are listed in table I with their description.

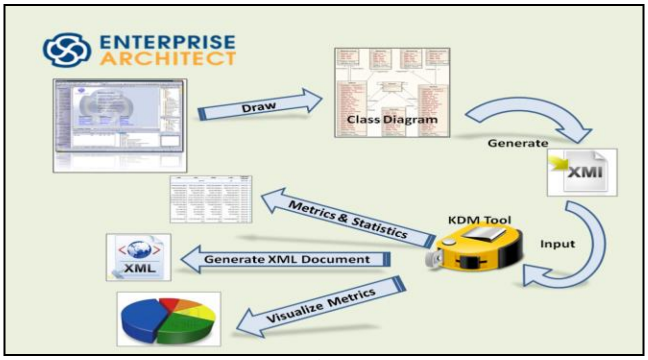

Figure 2. Input and output for KDM tool (All inside EA)

TABLE I. XMI TAGS USED IN THIS PAPER

Tag	Description
<UML:Class>	This tag is used to represent the class element. **UML:** is a namespace (Namespace provides a means to distinguish one XML vocabulary from another, which enables to create richer documents by combining multiple vocabularies into one document type [8]) which stands for "**omg.org/UML1.3**"
<UML:Attribute>	This tag is used to represent the attribute of the class
<UML:Operation>	This tag is used to represent the methods of the class
<UML:TaggedValue>	Tagged Values are a way of adding additional information to an element
<UML:Generalization>	This tag is used to represent inheritance relationship and it has two tagged values: "**ea_sourceName**" which represents source class (sub class) that inherits form target class (super class). "**ea_targetName**" which represents target class (super class) in which subclass inherits from it.
<UML:Association>	This tag is used to represent association, aggregation and composition. We can tell the difference between them by their tagged values. It has two tagged values for the source and the target classes.

- **XMI Parser**

XMI parser is used to extract data from XMI document, especially those tags listed in table I. Two important programming technologies were used in building XMI parser. They are: LINQ (Language Integrated Query) -to-XML and Lambda expressions. XMI parser will store all values of tags in lists like a list which contains the names for all classes, operations for each class, source classes and target classes for generalization relationship, ...etc. In order to find the name for the classes, attributes or operations in the XMI document, the following algorithm can be used:

Algorithm:

Step 1: Read XMI document and load it into XDocument object

Step 2: Determine the tag = "Class"

Step 3: Repeat for each tag

Step 3-1: Extract the value of the *name* attribute of the tag

Step 3-2: Save the name in the class list

Step 3-3: If not finish reading all tags, go to Step 3

Step 4: Display the class list

"Class list" will contain the name of each class in the XMI document; this list is the basis for all other methods in the XMI parser, because in order to find the name of each method in some class, it is needed to know the class name first (to which class they belong). In addition, to find attributes or methods names, the same algorithm can be used except for the tag which can be either as "Attribute" or "Operation".

In case of inheritance relationship, when it is needed to find the source classes (sub classes) and target classes in generalization relationship, the following algorithm can be used:

Algorithm:

Step 1: Read XMI document and load it into XDocument object

Step 2: Determine the relation = "Generalization"

Step 3: Repeat for each tag

Step 3-1: Extract the value of the ea_source tag attribute of the relation tag

Step 3-2: Save the name in a source list

Step 3-3: Extract the value of the ea_target tag attribute of the relation tag

Step 3-4: Save the name in a target list

Step 3-5: If not finish reading all tags, go to Step 3

Step 4: Display the list

"Source list" and "target list" contain the subtype classes and supertype classes in XMI document. By knowing the source and target classes in the generalization relationship, this will help calculating metrics like **MIF or AIF** which are related to inheritance concept. Thus, to find the source and the target list of another relation, only tag relation will change. Now consider the following Fig. 3 which represents a simple class diagram for aircraft types.

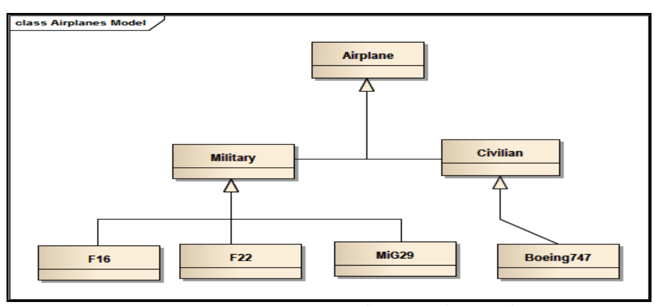

Figure 3. Simple class diagram for aircraft classification

According to the above algorithms, "class list" will contain the names for all classes, see table II, Source list" and "target list" for generalization relationship can be seen in table III. After collecting all required information, it is time to calculate the metrics for that design.

TABLE II. SAMPLE OF CLASS LIST

Classes Name
Airplane
Military
Civilian
Boeing747
MiG29
F16
F22

TABLE III SAMPLE OF SOURCE AND TARGET LISTS

Source Classes	Target Classes
Military	**Airplane**
Civilian	**Airplane**
Boeing747	**Civilian**
F16	**Military**
F22	**Military**
MiG29	**Military**

3. Suggested Algorithms For Mood Model

The person who sets the MOOD metrics was Fernando B. Abreu [1]. MOOD refers to a structural model of the object oriented paradigm like *encapsulation* as (Method Hiding Factor (MHF) and Attribute Hiding Factor (AHF)), *inheritance* as (Method Inheritance Factor (MIF) and Attribute Inheritance Factor (AHF)), *polymorphism* as (Polymorphism Factor (POF)), and *message passing* as (Coupling Factor (CF)). Each of the metrics was expressed to measure where the numerator defined the actual use of any one of the feature for a particular design. In MOOD model, there are two main features, namely methods and attributes [3].

Attributes are used to represent the status of object in the system and methods are used to maintain or modify several kinds of status of the objects [21][3].

MOOD metrics are designed to meet a particular set of criteria. They were also proposed by the MOOD project team. MOOD model in detail that will help to explain how to calculate each equation of MOOD model.

Algorithm for MHF Metric

MHF metric is used to measure encapsulation for the class diagram, actually for the invisibilities of methods for that class.

Algorithm:

Step 1: Import and verify XMI document (verification means it is XMI document)
Step 2: Parse XMI document
Step 3: Define a list for each access modifier of the methods
Step 4: Repeat for each class
Step 4-1: Store methods for each class; where the public methods are stored in the public list, private methods in private list and protected methods (if existed) in the protected list
Step 4-2: Go to step4
Step 5: Calculate MHF equation

$$MHF = \frac{\sum_{i=1}^{Tc}[\sum_{m=1}^{Md(Ci)}(1-V(Mmi))]}{\sum_{i=1}^{Tc}Md(Ci)} \quad \ldots\ldots (1)[19]$$

Where: TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). Md (Ci) = the number of methods defined in class Ci. V (Mmi) = Visibility value of a member (method or attribute), i.e. a value between 0-1 where public members = 1, private members = 0, and semi-public (e.g. protected) members are calculated as the number of classes that can access the member / total classes in the system (if working with different packages at the same time then the protected member is calculated. Otherwise it is considered the same as public in which it is equal to 1).
Step 6: Display MHF for the design

Algorithm for AHF Metric

AHF metric is used to measure encapsulation for the class diagram, actually for the invisibilities of the attributes for that class.

Algorithm:

Step 1: Import and verify XMI document
Step 2: Parse XMI document
Step 3: Define a list for each access modifier of the attributes
Step 4: Repeat for each class
Step 4-1: Store attributes for each class; where public attributes are stored in the public list, private attributes in private list and protected attributes (if existed) in the protected list
Step 4-2: Go to step4
Step 5: Calculate AHF equation

$$AHF = \frac{\sum_{i=1}^{Tc}[\sum_{m=1}^{Ad(Ci)}(1-V(Ami))]}{\sum_{i=1}^{Tc}Ad(Ci)} \quad \ldots\ldots\ldots (2)[19]$$

Where: TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). Ad (Ci) = the number of attributes defined in class Ci . V(Ami) is the same as V(Mmi) except it is for the attribute not for the method
Step 6: Display AHF for the design

A Suggested Algorithm for Finding the Root of Generalization or Aggregation Relationship

Sometimes a number of either a generalization hierarchy or aggregation hierarchy exists. This means that there are a number of roots in the design. In order to find the root of either of them, the following algorithm is suggested.

http://sites.google.com/site/ijcsis/
ISSN 1947-5500

Algorithm:

Step 1: Import and verify XMI document
Step 2: Parse XMI document
Step 3: Determine the type of the relationship
Step 4: Define lists for root classes, subclasses, and super classes.
Step 5: Repeat for each class in the list of super classes
Step 5-1: If any class is not in the list of source (subclasses), it means that the class does not inherit from other classes, so it is a root, add it to root list
Step 5-2: Go to Step 5
Step 6: If some class is repeated more than once, then delete it.
Step 7: Display root list

Algorithm for MIF Metric

MIF metric is used to measure the inheritance of the class diagram, which is the ratio of the inherited methods in it.

Algorithm:

Step 1: Import and verify XMI document
Step 2: Parse XMI document
Step 3: Define a list of source classes (subclasses) and another list of the target classes (super classes).
Step 4: Find the root of the generalization relationship
Step 5: Repeat for each class in source and target lists
Step 5-1: Store inherited methods in a list called inherited list
Step 5-2: Go to 5
Step 6: Calculate the equation of MIF

$$MIF = \frac{\sum_{i=1}^{TC} Mi(Ci)}{\sum_{i=1}^{TC} Ma(Ci)} \quad \text{.......... (3) [19]}$$

Where: TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). Mi is the number of inherited methods in Ci Ma is the number of available methods defined in Ci Md is the number of declared methods and not inherited in Ci Ma = Md + Mi of class Ci

Step 7: Display MIF for the design

Algorithm for AIF Metric

AIF metric is used to measure the inheritance of the class diagram, which is the ratio of the inherited attributes in it.

Algorithm:

Step 1: Import and verify XMI document
Step 2: Parse XMI document
Step 3: Define a list for source classes (subclasses) and another list for the target classes (super classes).
Step 4: Find the root of the generalization relationship
Step 5: Repeat for each class in source and target lists
Step 5-1: Store inherited attributes in a list called inherited list
Step 5-2: Go to Step 5
Step 6: Calculate the equation of AIF

$$AIF = \frac{\sum_{i=1}^{TC} Ai(Ci)}{\sum_{i=1}^{TC} Aa(Ci)} \quad \text{.......... (4) [19]}$$

Where:TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). Ai is the number of inherited Attributes in Class Ci. Aa is the number of available attributes defined in class Ci. Ad is the number of attributes declared in the class Ci. Aa = Ad + Ai of class Ci.

Step 7: Display AIF for the design

Algorithm for POF

POF measures the polymorphism of the class diagrams. This metric calculates the ratio of the polymorphic methods (degree of overriding in class diagram).

Algorithm:

Step 1: Import and verify XMI
Step 2: Parse XMI document
Step 3: Calculate the source and target classes
Step 4: NC = total number of classes
Step 5: Repeat for each class while < NC
Step 5-1: Find the descendant classes for each class in the target list
Step 5-2: Find the new (declared) method for each class and put them in a list
Step 5-3: Find the overridden methods and put them in a list
Step 5-4: Go to Step 5
Step 6: Calculate POF

$$POF = \frac{\sum_{i=1}^{TC} Mo(Ci)}{\sum_{i=1}^{TC} [Mn(Ci) * DC(Ci)]} \quad \text{........... (5) [19]}$$

Where: TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). Mo(Ci) is the overridden methods for Class Ci. Mn(Ci) is the new methods defined in Class Ci. DC(Ci) is the descendant counts (number of subclasses) for Class Ci .

Step 7: Display POF for the design

Algorithm for CF

CF is used to measure the coupling of the class diagram when one class calls a method of another class, then they are coupled.

Algorithm:

Step 1: Import and verify XMI
Step 2: Parse XMI document
Step 3: Find the source and target classes of the association relationship
Step 4: Concatenate the target list with the source list, remove duplication and put it into a new list called c list
Step 5: Repeat for each class in c list
Step 5-1: If a class has any relationship but not generalization then put it into a list
Step 5-2: Go to Step 5
Step 6: Apply CF equation

$$CF = \frac{\sum_{i=1}^{TC} [\sum_{j=1}^{TC} is\_Client\,(Ci,Cj)]}{TC^2 - TC} \quad \text{......... (6) [19]}$$

Here: TC = total number of classes. Summation occurs over i=1 to TC. Ci = class with index i (current class). is_client(Ci,Cj)=1 if Ci contains at least one non inheritance reference to a method or attribute of a class and Cj=0 otherwise.

http://sites.google.com/site/ijcsis/
ISSN 1947-5500

Step 7: Display CF

Now, after all algorithms about MOOD model are explained, consider the following example which illustrates all algorithms above, see Fig. 4. Now consider the table IV which represents the class diagram as numbers.

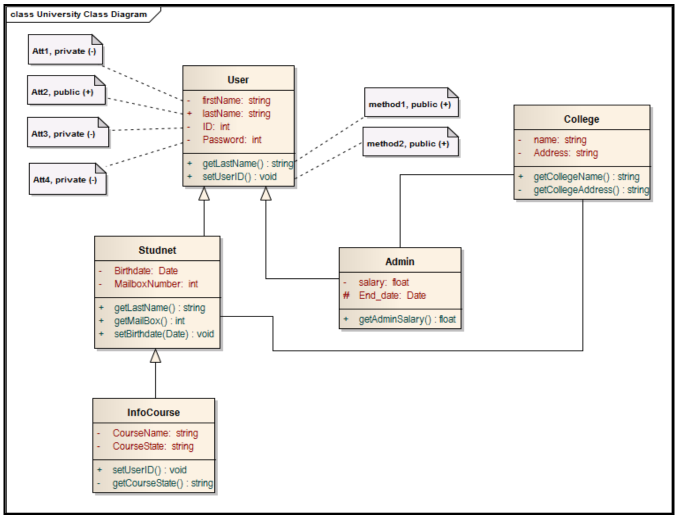

Figure 4. Simple class diagram for some of university emtities

TABLE IV CLASS DIAGRAM ANALYSIS

Class	Att.	Method	+ Att.	- Att.	#Att.	+M.	- M.	#M.
User	4	2	1	3	0	2	0	0
Student	2	3	0	2	0	3	0	0
Admin	2	1	0	1	1	1	0	0
InfoCourse	2	2	0	2	0	1	1	0
College	2	2	0	2	0	1	1	0

Where: Att. is an abbreviation for attribute. M. is an abbreviation for method. + prefix means public modifier. - prefix means private modifier. # prefix means protected modifier.

Fig. 4 shows a simple class diagram with 5 classes, 12 attributes and 10 methods, to calculate the metrics according to the table above using metrics equations.

1.Encapsulation (Eq. (1) and Eq. (2))

$$MHF = \frac{0+1+0+0+1}{2+2+1+3+2} = \frac{2}{10} = 20\ \%$$

$$AHF = \frac{3+2+1+2+2}{4+2+2+2+2} = \frac{10}{12} = 83.33\ \%$$

2. Inheritance (Eq. (3) and Eq. (4))

$$MIF = \frac{0+2+2+5}{2+5+3+7} = \frac{9}{17} = 52.94\%$$

$$AIF = \frac{0+4+4+6}{4+6+6+8} = \frac{14}{24} = 58.33\%$$

3.Polymorphism (Eq. (5))

$$POF = \frac{0+1+0+1}{4+2+0+0} = \frac{2}{6} = 33.33\%$$

4.Coupling (Eq. (6))

$$CF = \frac{3}{25-5} = \frac{3}{20} = 15\%$$

It is concluded from table V, that AHF, MIF, AIF are within the limit while MHF, POF, CF are not within the standard limit. So, a correction or a review of the design is needed.

TABLE V Standard intervals for mood model [1]

Metrics	Minimum Value	Maximum Value
MHF	12.7%	21.8%
AHF	75.2%	100%
MIF	66.4%	78.5%
AIF	52.7%	66.3%
POF	2.7%	9.6%
CF	4.0%	11.2%

4. Suggested Algorithms for MEMOOD model

The ever changing world makes maintainability a strong quality requirement for the majority of software systems. The maintainability measurement during the development phases of object oriented system estimates the maintenance effort. It also evaluates the likelihood that the software product will be easy to maintain. Despite the fact that software maintenance is an expensive and challenging task, it is not properly managed and often ignored. One reason for this poor management is the lack of proven measures for software maintainability [18].

- **Algorithm for Maintainability**

Maintainability is defined as "the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment" [9]. As class diagrams play a key role in the design phase of object-oriented software, early estimation of their maintainability may help designers to incorporate required enhancements and corrections in order to improve their maintainability and consequently the maintainability of the final software to be delivered in future. Two quality attributes of class diagram, namely understandability and modifiability are focused to estimate their maintainability [18].

Maintainability means how easy it is for software engineer to maintain the design by means of adapting, correcting or improving the design. In order to calculate the maintainability, understandability and modifiability are used along with a number of constants to form the equation (see Eq.(7)).

Algorithm:

Step 1: Import XMI
Step 2: Parse XMI document
Step 3: Calculate understandability and modifiability
Step 4: Apply maintainability model

$$Maintainability = -0.126 + 0.645 * Understadability + 0.502 * Modifiability \quad \ldots\ldots (7)$$

Step 5: Display maintainability

- **Algorithm for Understandability**

Understandability means how much the software engineer understands the design that he is working on or how easy to comprehend it. In order to calculate the understandability of the design it is needed first to find two metrics, named NC and NGenH (see table VI). These two metrics along with some constant numbers are used to calculate the understandability of the design.

Algorithm:

Step 1: Import XMI
Step 2: Parse XMI document
Step 3: Calculate NC and NgenH for the design
Step 4: Apply understandability

$$Understandability = 1.166 + 0.256 * NC - 0.394 * NGenH \ldots\ldots (8)$$

Where: NC is the total number of classes. NGenH is the number of generalization hierarchies.

Step 5: Save the value of understandability
Step 6: Display understandability

- **Algorithm for Modifiability**

Modifiability means the ability of software engineer to modify the design without affecting it. In order to calculate the modifiability of the design it is needed first to find five metrics, named NC, Ngen, NgenH, NaggH and MaxDIT (see table VI). These five metrics along with some constant numbers are used to calculate the modifiability of the design.

Algorithm:

Step 1: Import XMI
Step 2: Parse XMI document
Step 3: Calculate NC, Ngen, NgenH, NaggH and MaxDIT for the design
Step 4: Apply modifiability equation

$$Modifiability = 0.629 + 0.471 * NC - 0.173 * NGen - 0.616 * NAggH - 0.696 * NGenH + 0.396 * MaxDIT$$

………. (9)[18]

Where: NC is the total number of classes. NGen is the number of generalization relationship (inheritance relationship between super class and sub class). NAggH is the number of aggregation relationship hierarchies. NGenH is the number of generalization hierarchies in the design. MaxDIT is the maximum depth of the inheritance in the design

Step 5: Save the value of modifiability
Step 6: Display modifiability

Metrics in table VI have been selected for quantifying understandability and modifiability of class diagram. It had already been empirically validated that these metrics are correlated with understandability and modifiability of class diagram [27][4].

In order to calculate the Maintainability Estimation model see Fig.5 [18], Both the Understandability and the Modifiability of the design are used. *Understandability* in our context means the extent of users (software engineer or programmer) capability with different backgrounds to understand the software design. Understandability of the design can be calculated as in Eq. (10)[18].

$$Understandability = 1.166 + 0.256 * NC - 0.394 * NGenH \ \ldots..(10)$$

TABLE VI SIZA AND CTRUCTURAL COMPLEXITY METRICS FOR UML CLASS DIAGRAM

Metric Name	**Metrics Definition**
Number of classes (NC)	The total number of classes
Number of attributes (NA)	The total number of attributes
Number of methods (NM)	The total number of methods
Number of associations (NAssoc)	The total number of associations
Number of aggregation (NAgg)	The total number of aggregation relationships within a class diagram (each whole-part pair in an aggregation relationship)
Number of dependencies (NDep)	The total number of dependency relationships
Number of generalizations (NGen)	The total number of generalization relationships within a class diagram (each parent-child pair in a generalization relationship)
Number of aggregations Hierarchies (NAggH)	The total number of aggregation hierarchies (whole-part structures) within a class diagram
Number of generalizations Hierarchies (NGenH)	The total number of generalization hierarchies within a class diagram
Maximum depth of inheritance (MaxDIT)	It is the maximum of the DIT (Depth of Inheritance Tree) values obtained for each class of the class diagram. The DIT value for a class within a generalization hierarchy is the longest path from the class to the root of the hierarchy
Maximum aggregation hierarchy (MaxHAgg)	It is the maximum of the HAgg values obtained for each class of the class diagram. The HAgg value for a class within an aggregation hierarchy is the longest path from the class to the Leaves.

Modifiability in our context is the capability to modify the design without affecting the overall system. See equation (11)[18].

$$Modifiability = 0.629 + 0.471 * NC - 0.173 * NGen - 0.616 * NAggH - 0.696 * NGenH + 0.396 * MaxDIT \ \ldots\ldots(11)$$

After calculating understandability and modifiability quality attributes it is possible now to find the maintainability of software design. See equation (12)[18].

$$Maintainability = -0.126 + 0.645 * Understadability + 0.502 * Modifiability \ \ldots.(12)$$

The values of understandability, modifiability and maintainability are of immediate use in the software development process. These values may help software designers to review the design and take appropriate corrective measures, early in the development life cycle, in order to control or at least reduce future maintenance cost [18].

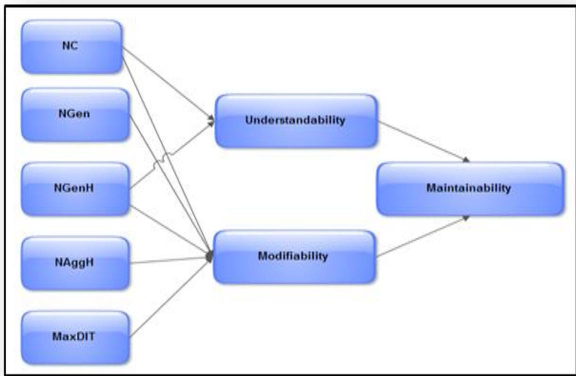

Figure 5. Maintainbility Estimation Model (MEMOOD) [18]

Now go back to Fig.4, the following table VII can be deduced.

TABLE VII METRICS USED TO CALCULATE MEMOOD MODEL, ALSO SEE TABLE VI

NC	Ngen	NgenH	NaggH	MaxDit
5	3	1	0	2

Where

- NC is the total number of classes = 5.
- Ngen is the number of generalization relationships = 3.
- NgenH is the number of generalization hierarchy =1, since there is only one generalization tree.
- MaxDit is the maximum number of depth of inheritance tree =2, since the User class is in level 0 of the generalization hierarchy, Student and Admin classes are in level 1, and InfoCourse class is in level 2.
- NaggH is the number of the aggregation hierarchy = 0, since there is no aggregation hierarchy.

1. **Understandability (Eq. 10)**
   Understandability = 1.166 + 0.256*5 – 0.394*1 = 2.05
2. **Modifiability (Eq. 11)**
   Modifiability = 0.629 + 0.471*5 – 0.173*3 – 0.616*0 –0.696*1+ 0.396*2 = 2.56
3. **Maintainability (Eq. 12)**
   Maintainability = -0.126 + 0.645* 2.05 + 0.502*2.56 = 2.48

5. XML

XML is a standard technology that is concerned with the description and structuring of data by means of **tags** that are similar to HTML ones. XML can be used almost in every application especially in the web. See Fig. 6 which represents a sample of XML. It can be seen from the figure above that XML is used to describe a book; its title, author, price, etc. XML sometimes is used as intermediate data that flow between applications and these applications passes these XML between each other, so XML can be used as a bridge between various applications. Going back to Fig. 2, it can be seen that XML was used as **output** from KDM tool which is the main tool, KRS and KDB are developed to support it. So, how can these tools communicate between each other? The answer is by using XML as a bridge between them. XML parser was built for that XML in which it will be understood and used properly. A specific structure of XML is proposed in this paper (see table VIII).

TABLE VIII XML STRUCTURE OF THE KDM XML OUTPUT

Tag	Description
<Metrics>	This tag is used as root for a number of metrics
<Metric Id=”” />	This tag is used as an identifier for the metrics
<DesignerName>	This tag is used to describe the designer name
<ModelName>	This tag is used to describe the model name
<MHF>	This tag is used to describe the MHF metric
<AHF>	This tag is used to describe the AHF metric
<MIF>	This tag is used to describe the MIF metric
<AIF>	This tag is used to describe the AIF metric
<CF>	This tag is used to describe the CF metric
<POF>	This tag is used to describe the POF metric
<Understandability>	This tag is used to describe the understandability
<Modifiability>	This tag is used to describe the modifiability
<Maintainability>	This tag is used to describe the maintainability

A sample of XML document can be seen in Fig. 7 which is also the XML output from KDM tool.

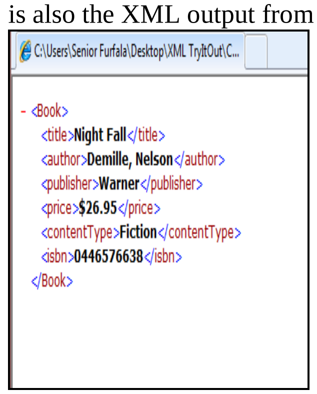

Figure 6. XML sample

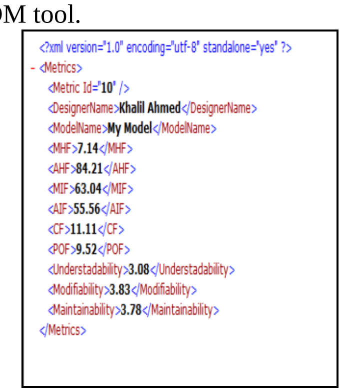

Figure 7. XML output from KDM tool

6. KDM Tool Sequence Diagram

KDM tool sequence of operations starts after importing XMI document and ends with exporting XML, see Fig. 8

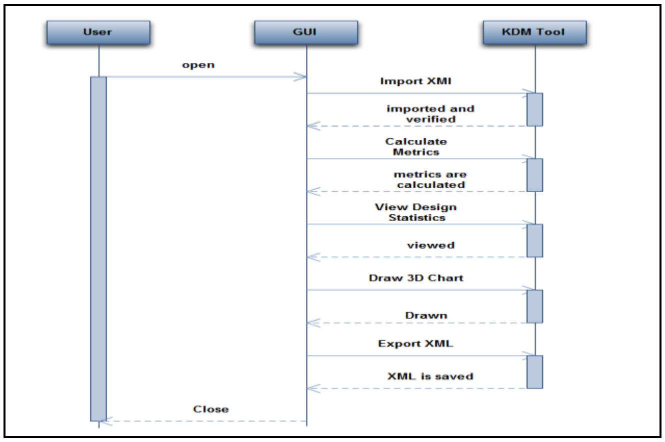

Figure 8. KDM tool sequence diagram

B. KRS (K Reporting Service) Tool

KRS tool is a reporting tool that is integrated with EA. The purpose of this tool is to document metrics as a report for a project manager or maybe for the team leader. It is said earlier that KRS tool supports KDM tool and they communicate by exchanging XML. The **input** for KRS tool is the XML output of KDM tool. So, KRS has a parser for the XML generated by KDM tool. The **output** of KRS tool is a crystal report which contains metrics and two graphs, see Fig.9.

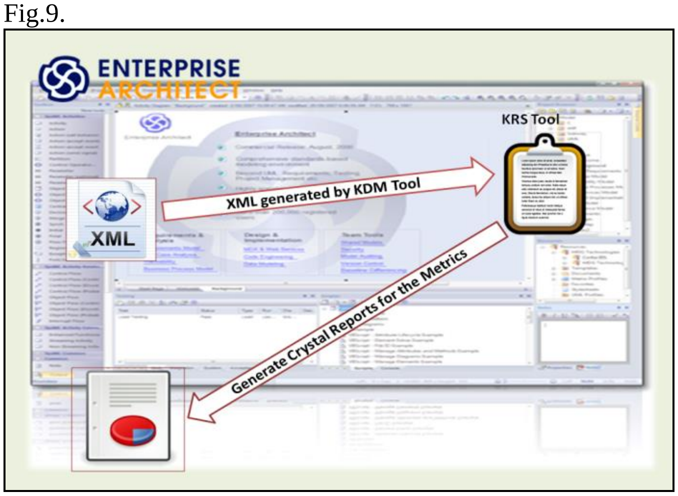

Figure 9. Input and output for KRS Tool

1. How KRS Tool Works

Before discussing how it works, it is needed to know where KRS tool works, and in which phase it supports in SDLC. KDM tool is an Upper CASE tool and since KRS works with the documentation of metrics in the same phase, it is deduced that KRS tool is also an Upper CASE tool. KRS tool accepts XML document that is generated by KDM as input, see Fig. 7. Then XML document proceeds to XML parser which extracts the information and prepares it to be fed into the report generator and produces a crystal report of the design metrics. See Fig. 10.

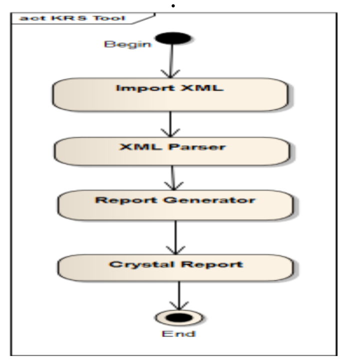

Figure 10. KRS Tool Workflow

2. XML Parser

XML parser extracts the value of each tag listed in table VIII from the XML document which is depicted in Fig. 11.

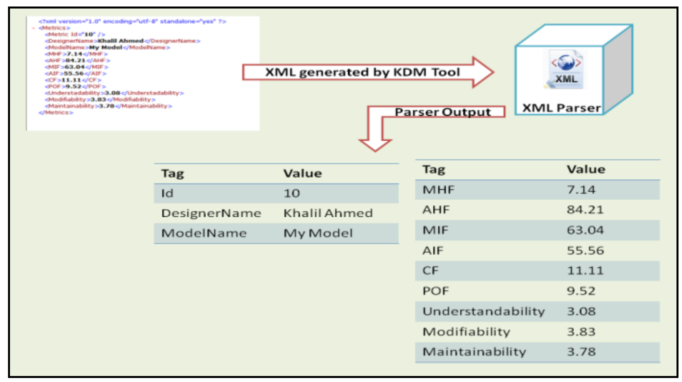

Figure 11. XML Parser

Algorithm:

Step 1: Import and verify XML

Step 2: Extract model name and designer name from XML document and put them in a list.

Step 3: Extract all metrics and put them in a list

Step 4: End

C. KDB (K Database) Tool

KDB tool is a database tool that is integrated with EA; KDB tool is used to store metrics in a database; may be for checking the metrics against another system (design) which has similar requirements or used as a historical data. KDB tool has the same XML parser of KRS tool. KDB tool accepts the same XML which is generated by KDM tool as **input**, and stores the numeric (double data type) value of metrics in the database

1. How KDB Tool Works

Any tool that supports any phase in SDLC is considered a CASE tool, otherwise it is not. Since KDB tool stores metrics which are software engineering information and supports KDM tool, so, as a result, it is a CASE tool and can be considered an Upper CASE tool. KDB tool takes XML document which is generated by KDM tool, parses it, and formats metrics in a way that can be stored in the database.

http://sites.google.com/site/ijcsis/
ISSN 1947-5500

VI. Testing the Proposed Tools

A case study has been taken from [5] and modified so that all metrics can be calculated. The case study is about a Student Registration System at university. By using this system, students have access to the information of the available courses, and they can also register in the system, it is managed by a special user who is allowed to modify the required courses in the catalogue. This system was modeled using EA v7.5. See Fig. 12 which represents the class diagram for the system. This class diagram is exported from EA as XMI which will be the input for KDM tool, and by pressing on Metrics button the metrics are calculated. See Fig. 13.

It can be seen that class names group box is filled with all classes from the class diagram. MOOD and MEMOOD values are calculated. Design statistics can be seen in Fig. 14.

Form the statistics above, it can be seen that there are 9 classes, 19 attributes, 28 methods with 9 relations, one aggregation hierarchy, one generalization hierarchy and the maximum depth of inheritance is 2. Required information group box is used to export XML document that contains the metrics along with the model name, designer name and model id. This document is used as input for KRS and KDB tool. Visualization of metrics for example can be seen in Fig. 15 where red color means that the design is needed to be reviewed according to metric value, green color means that the metric value is within the allowed range and no review is needed.

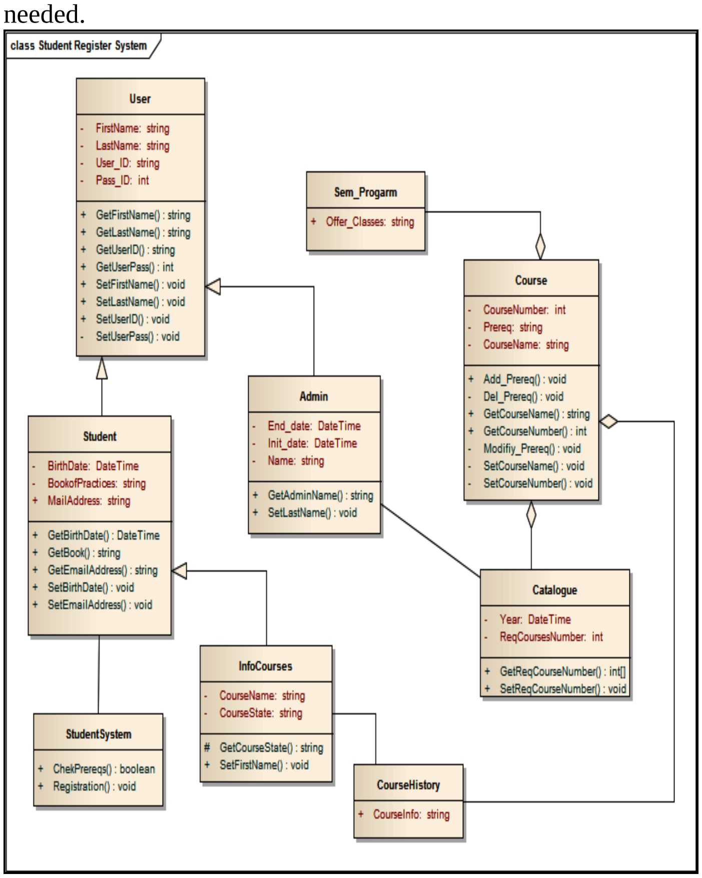

Figure 12.Student Registration System Class Diagram

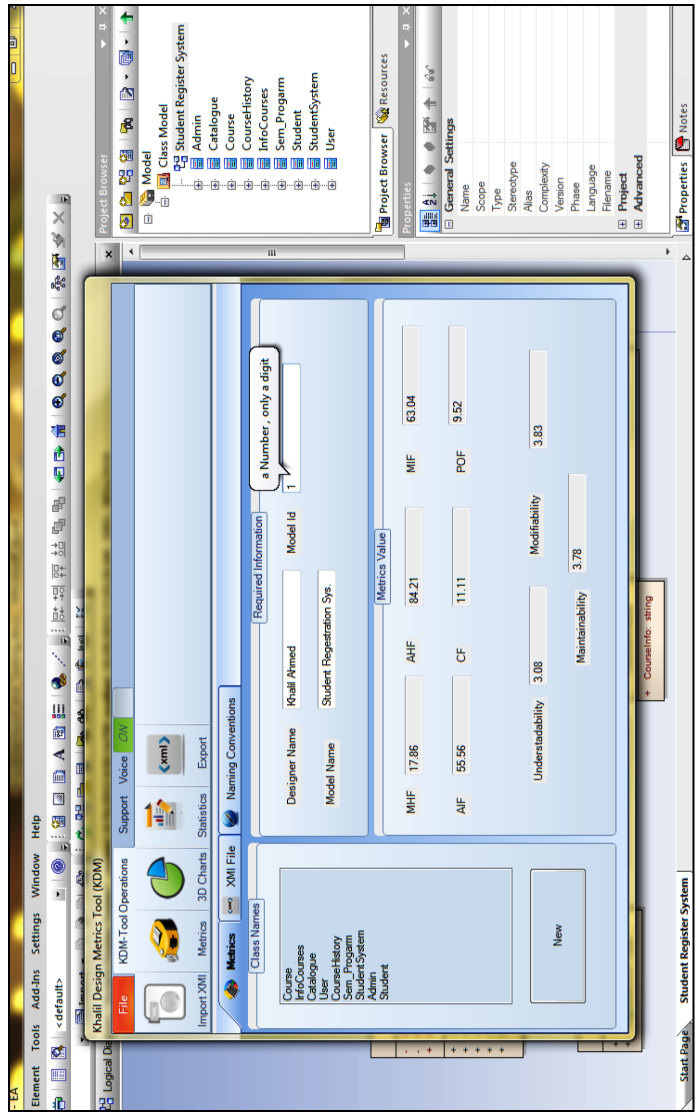

Figure 13. Metrics for Student Registrations System.

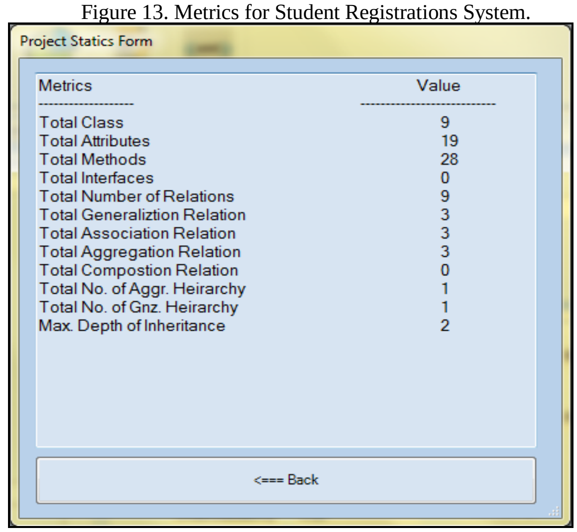

Figure 14. Design Statistics for the Class Diagram.

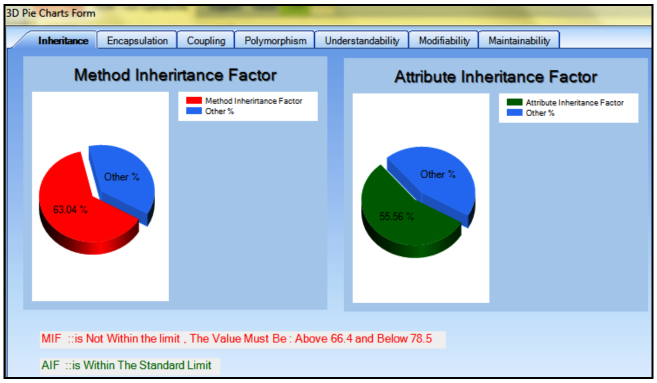

Figure 15. MIF and AIF Metrics 3D-Pie Chart.

KDM tool supports JAVA design naming conventions of the design. See Fig. 16.

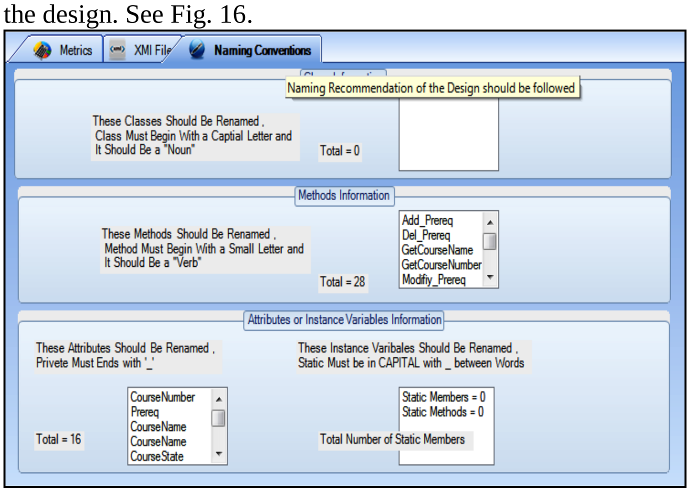

Figure 16. Naming Conventions.

By opening KRS tool and importing the XML document generated by KDM tool, the output is a crystal report. See Fig. 17.

Figure 17. A Crystal Report for the Metrics of the System.

By opening KDB tool and importing XML document generated by KDM tool, XML parser will extract metrics and KDB tool will load them into the text boxes and into the XML tab . By pressing on View All Data button, a new form will open and it will contain the metrics that are stored in the database. See Fig 18.

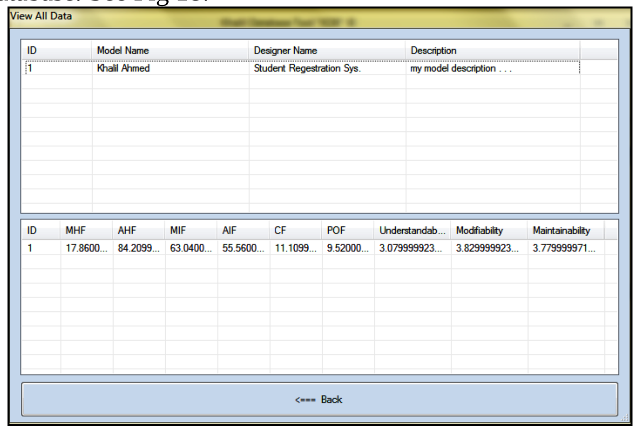

Figure 18. View All Data Form.

VII. DISCUSSION OF TESTING RESULTS

KDM tool has succeeded in calculating MOOD and MEMOOD metrics and it gives 100% correct results, because the metrics are calculated by hand and have the same values of KDM tool. From the results of KDM tool, table IX can be deduced.

TABLE IX Metrics discussion

Metric	Recommendation	Value	Within the limit	Outside the limit
MHF	No recommendation is needed	17.86	✓	-
AHF	No recommendation is needed	84.21	✓	-
MIF	**It is recommended that the number of inherited methods in the design should be reduced**	**63.04**	-	✓
AIF	No recommendation is needed	55.56	✓	-
CF	No recommendation is needed	11.11	✓	-
POF	No recommendation is needed	9.52	✓	-
Understandability	No recommendation is needed	3.08	✓	-
Modifiability	No recommendation is needed	3.83	✓	-
Maintainability	No recommendation is needed	3.78	✓	-

From the table above, it can be concluded that the design is fine and a review must be taken for MIF value.

A. Evaluation of the Proposed Tools

In this paper questionnaire as in [31] has been conducted by a twenty person who are considered as users of the proposed tools (programmers and software engineers). The samples were taken from people within the field (Computer Science and Software Engineering). The questionnaire divided into four sections namely:

1. Evaluating the tools generally,
2. Evaluating KDM Tool,
3. Evaluating KRS Tool, and
4. Evaluating KDB Tool.

Using SPSS program to get the results, see the table below.

TABLE X QUESTIONNAIRE RESULTS

Tool Name	Questionnaire Result
Evaluating the tools generally	94.4
Evaluating KDM Tool	93.8
Evaluating KRS Tool	96.6
Evaluating KDB Tool	90.7

VIII. CONCLUSION

Through the building and testing of KDM, KRS, and KDB tools, conclusions were that; KDM tool accepts XMI or XML documents generated by EA since EA export UML diagram as .XML or .XMI extension. Documentation of metrics do helps project managers or team leaders to monitor the progress by using KRS tool. Storage of metrics can help designers to compare the metrics of some system with others. So, it can be used as a historical data by using KDB tool. MOOD model help to identify problems of the design by means of metrics that uses the OO concepts which allow software engineers to early access software design and yet improve it. MEMOOD model calculates understandability, modifiability, and maintainability of the design which are vital to know early in design phase. Without XMI, no UML diagram can be described. XML can be used as a bridge between tools or as intermediate data. Generics in C# (Lists) are really important due to their dynamic allocation. When EA does not support database or reports as add-in, integration must be used.

Future works can be summarized as the follows: Developing an add-in for ArgoUML and StarUML to calculate metrics since they also do not support metrics for the design. Evolving KDM tool to take not just XMI or XML as input but also the source code of Java, C# and C++.

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[1] Abreu, F.B., (1995), "Design Metrics for Object-Oriented Software Systems", ECOOP'95 Quantitative Methods Workshop in Aarhus, August

[2] Ahmed, S.H. , Soliman, T.H.A., and Sewisy, A.A., (2013) , "A Hybrid Metrics Suite for Evaluating Object-Oriented Design" , International Journal of Software Engineering , Vol.6 , Issue-1, pp: 65-82, ISSN: 16876954

[3] Al-Zobaidy, L. M. and Ibrahim, K. A. , (2012) , "Existing Object Oriented Design Metrics a Study and a Comparison" , published in the 5th Scientfic Conference in Information Technology (CCIT), Computer Science and Mathematics ,Dec 19-20, University of Mosul , Iraq

[4] Calero, C., Genero, M., and Piattini,M. , (2002), "Empirical Validation of Class Diagram Metrics" ,Journal of Empirical Software Engineering, Proceedings of the International Symposium on Empirical Software Engineering (ISESE'02) , pp:195-203, ISBN: 07695179, IEEE

[5] Genero, M., Piattini, M., and Calero, C., (2002), "An Empirical Study to Validate Metrics for Class Diagrams", Department of Computer Science,University of Castilla-La Mancha, Spain

[6] Girgis, M.R., Mahmoud, T.M., and Nour, R.R, (2009) , "UML Class Diagram Metrics Tool", International Conference on Computer Engineering & Systems , pp: 423-428, IEEE

[7] Hilera, J.R., and Fernández, L., (2012), "A Web Service for Calculating the Metrics of UML Class Diagrams" , Dobb's Special Report June , www.drdobbs.com, The world of software development

[8] Hunter, D., Rafter, J., Fawcett, J., Vlist, E., Ayers, D., Duckett, J., Watt, A., and McKinnon, L., (2007), "Beginning XML", 4th Edition, Wiley Publishing , ISBN:978-0-470-11487-2

[9] IEEE Standard Glossary of Software Engineering Terminology, (1990), IEEE-Std 610.12 , ISBN 1-55937-067

[10] Jassim, F. , and Altaani, F. , (2013) , "Statistical Approach for Predicting Factors of Mood Method for Object Oriented", Uinversity of Irbid, Irbid, Jordan , arXiv preprint arXiv:1302.5454.

[11] Mago, J. , and Kaur, P., (2012) , "Analysis of Quality of the Design of the Object Oriented Software using Fuzzy Logic", International Journal of Computer Applications (IJCA) , Vol. iRAFIT Issue: 3 , pp : 21-25

[12] Mehra, S., and Maini, R., (2011), "A METRIC FRAMEWORK FOR ANALYSIS OF OOD" , Journal of Global Research in Computer Science , Vol. 2 , No-7, pp: 67-70 , ISSN :2229-371X

[13] Paterson, T., Russell, C., and Dewar, R. , (2002), "Object-oriented software design metrics from XMI", School of Mathematical and Computer Science, Heriot-Watt University, Scotland

[14] Poornima, U.S. , (2011) , "A Quantitative Measure for Object Oriented Design Approach for Large-Scale Systems" , International Journal on Computer Science and Engineering (IJCSE) ,Vol. 3 No-9 , pp: 3237-3242, ISSN : 0975-3397

[15] Pressman, R., (2011) ,"Software Engineering: A Practitioner's Approach" 7th Edition, McGraw-Hill , ISBN :978–0–07–337597–7

[16] Rani, T., Sanyal, M., and Garg, S., (2012) , "Measuring Software Design Class Metrics:- A Tool Approach" , International Journal of Engineering Research & Technology (IJERT), Vol.1,Issue-7, ISSN: 2278-0181

[17] Reißing, R., (2001), "Assessing the Quality of Object-Oriented Designs", OOPSLA, Institute of Computer Science, University of Stuttgart , Germany

[18] Rizvi, S.W.A., Khan, R.A., (2010), "Maintainability Estimation Model for Object-Oriented Software in Design Phase (MEMOOD)" Journal of Computing, Vol. 2, Issue-4, ISSN: 2151-9617

[19] Sastry, J.S.V.R.S. , Ramesh, K.V., and Padmaja, M. , (2011), "Measuring Object-Oriented Systems Based On The Experimental Analysis Of The Complexity Metrics" , International Journal of Engineering Science and Technology (IJEST) , Vol. 3 , No. 5 , pp: 3726-3731, ISSN : 0975-5462

[20] Schach, S.R., (2011) ,"Object-Oriented and Classical Software Engineering" , 8th edition, McGraw Hill , ISBN: 978-0-07-337618-9

[21] Sharma, A., and Dubey, S.K., (2012), "Comparison of Software Quality Metrics for Oriented Oriented System", International Journal of Computer Science & Management Studies (IJCSMS), Vol. 12, June 2012 , ISSN:2231 –5268

[22] Sharma, A.K., Kalia, A. , and Singh, H. , (2012) , " Metrics Identification for Measuring Object Oriented Software Quality" ,International Journal of Soft Computing and Engineering (IJSCE), Vol. 2, Issue-5, November 2012 , ISSN: 2231-2307

[23] Sommerville, I. , (2011) , "Software Engineering" , 9th Edition , Addison Wesley, ISBN-13: 978-0-13-703515-1

[24] Sparx Systems, (2009) , "Enterprise Architect User Guide v7.5 " , Enterprise Architect Software , http://www.sparxsystems.com/products/ea/7.5/

[25] Sparx Systems, (2013) , "Enterprise Architect 10 Reviewer's Guide" , Sparx Systems Community , http://www.sparxsystems.com/resources/whitepapers/

[26] Summers, B.L., (2011), " Software Engineering Reviews and Audits", CRC Press , ISBN:978-1-4398-5145-6

[27] Web page ,Class diagram examples, 2012, http://www.programsformca.com/2012/03/uml-diagrams-for-hospital-mgmt-system

[28] Web page, http://www.sparxsystems.com/enterprise_architect_user_guide/9.3/index.html

[29] Web page, http://www.sparxsystems.com/enterprise-architect/index.html

[30] Website, http://www.sparxsystems.com/

[31] Al-Hadidi, K. A., Constructing a Tool for Measuring a Quality of Object Oriented Design to Enterprise Architect v7.5, MSc Thesis , Software Engineering, Mosul University, Collage of Computer sc. And Mathematic Mosul , Iraq.

[32] Kan, S.H., (2002) , "Metrics and Models in Software Quality Engineering" , 2nd Edition , Addison Wesley , ISBN:0-201-72915-6

[33] Bellekens, G. , (2011) , "Tutorial: Create your first C# Enterprise Architect add-in in 10 minutes" , Sparx System Community.

[34] Al-Zobaidy, L. M. and Ibrahim, K. A. , (2012) , "Existing Object Oriented Design Metrics a Study and a Comparison" , published in the 5th Scientfic Conference in Information Technology (CCIT), Computer Science and Mathematics ,Dec 19-20, University of Mosul , Iraq

[35] Ibrahim, K. A. , (2013), " Constructing a Tool for Measuring a Quality of Object Oriented Design to Enterprise Architect v7.5" , Master Thesis, College of Computer Sciences and Mathematics ,University of Mosul

AUTHORS PROFILE

First Author :**Dr. laheeb M. Alzubaidy**, have BSc. In 1987, MSc. In 1992 And PhD in 2002, in computer Sc. From Dept. of computer Sc, university of Mosul, Iraq. Associative professor in 2003, Head of Dept of Computer Sc. In 2003, visiting lecturer in Isra private university in 2004, head of Dept of Software Engineerinh in 2007, Visiting lecturer in USM university , NAV6 center in 2009, interested research fields are in Artificial Intelligent technique, network security, image processing , pattern recognition, software eng.

IJCSIS REVIEWERS' LIST

CALL FOR PAPERS
International Journal of Computer Science and Information Security

IJCSIS 2015

ISSN: 1947-5500

http://sites.google.com/site/ijcsis/

International Journal Computer Science and Information Security, IJCSIS, is the premier scholarly venue in the areas of computer science and security issues. IJCSIS 2011 will provide a high profile, leading edge platform for researchers and engineers alike to publish state-of-the-art research in the respective fields of information technology and communication security. The journal will feature a diverse mixture of publication articles including core and applied computer science related topics.

Authors are solicited to contribute to the special issue by submitting articles that illustrate research results, projects, surveying works and industrial experiences that describe significant advances in the following areas, but are not limited to. Submissions may span a broad range of topics, e.g.:

Track A: Security

Access control, Anonymity, Audit and audit reduction & Authentication and authorization, Applied cryptography, Cryptanalysis, Digital Signatures, Biometric security, Boundary control devices, Certification and accreditation, Cross-layer design for security, Security & Network Management, Data and system integrity, Database security, Defensive information warfare, Denial of service protection, Intrusion Detection, Anti-malware, Distributed systems security, Electronic commerce, E-mail security, Spam, Phishing, E-mail fraud, Virus, worms, Trojan Protection, Grid security, Information hiding and watermarking & Information survivability, Insider threat protection, Integrity
Intellectual property protection, Internet/Intranet Security, Key management and key recovery, Language-based security, Mobile and wireless security, Mobile, Ad Hoc and Sensor Network Security, Monitoring and surveillance, Multimedia security ,Operating system security, Peer-to-peer security, Performance Evaluations of Protocols & Security Application, Privacy and data protection, Product evaluation criteria and compliance, Risk evaluation and security certification, Risk/vulnerability assessment, Security & Network Management, Security Models & protocols, Security threats & countermeasures (DDoS, MiM, Session Hijacking, Replay attack etc,), Trusted computing, Ubiquitous Computing Security, Virtualization security, VoIP security, Web 2.0 security, Submission Procedures, Active Defense Systems, Adaptive Defense Systems, Benchmark, Analysis and Evaluation of Security Systems, Distributed Access Control and Trust Management, Distributed Attack Systems and Mechanisms, Distributed Intrusion Detection/Prevention Systems, Denial-of-Service Attacks and Countermeasures, High Performance Security Systems, Identity Management and Authentication, Implementation, Deployment and Management of Security Systems, Intelligent Defense Systems, Internet and Network Forensics, Large-scale Attacks and Defense, RFID Security and Privacy, Security Architectures in Distributed Network Systems, Security for Critical Infrastructures, Security for P2P systems and Grid Systems, Security in E-Commerce, Security and Privacy in Wireless Networks, Secure Mobile Agents and Mobile Code, Security Protocols, Security Simulation and Tools, Security Theory and Tools, Standards and Assurance Methods, Trusted Computing, Viruses, Worms, and Other Malicious Code, World Wide Web Security, Novel and emerging secure architecture, Study of attack strategies, attack modeling, Case studies and analysis of actual attacks, Continuity of Operations during an attack, Key management, Trust management, Intrusion detection techniques, Intrusion response, alarm management, and correlation analysis, Study of tradeoffs between security and system performance, Intrusion tolerance systems, Secure protocols, Security in wireless networks (e.g. mesh networks, sensor networks, etc.), Cryptography and Secure Communications, Computer Forensics, Recovery and Healing, Security Visualization, Formal Methods in Security, Principles for Designing a Secure Computing System, Autonomic Security, Internet Security, Security in Health Care Systems, Security Solutions Using Reconfigurable Computing, Adaptive and Intelligent Defense Systems, Authentication and Access control, Denial of service attacks and countermeasures, Identity, Route and

Location Anonymity schemes, Intrusion detection and prevention techniques, Cryptography, encryption algorithms and Key management schemes, Secure routing schemes, Secure neighbor discovery and localization, Trust establishment and maintenance, Confidentiality and data integrity, Security architectures, deployments and solutions, Emerging threats to cloud-based services, Security model for new services, Cloud-aware web service security, Information hiding in Cloud Computing, Securing distributed data storage in cloud, Security, privacy and trust in mobile computing systems and applications, **Middleware security & Security features:** middleware software is an asset on
its own and has to be protected, interaction between security-specific and other middleware features, e.g., context-awareness, **Middleware-level security monitoring and measurement:** metrics and mechanisms for quantification and evaluation of security enforced by the middleware, **Security co-design:** trade-off and co-design between application-based and middleware-based security, **Policy-based management:** innovative support for policy-based definition and enforcement of security concerns, **Identification and authentication mechanisms:** Means to capture application specific constraints in defining and enforcing access control rules, **Middleware-oriented security patterns:** identification of patterns for sound, reusable security, **Security in aspect-based middleware:** mechanisms for isolating and enforcing security aspects, **Security in agent-based platforms:** protection for mobile code and platforms, Smart Devices: Biometrics, National ID cards, Embedded Systems Security and TPMs, RFID Systems Security, Smart Card Security, Pervasive Systems: Digital Rights Management (DRM) in pervasive environments, Intrusion Detection and Information Filtering, Localization Systems Security (Tracking of People and Goods), Mobile Commerce Security, Privacy Enhancing Technologies, Security Protocols (for Identification and Authentication, Confidentiality and Privacy, and Integrity), Ubiquitous Networks: Ad Hoc Networks Security, Delay-Tolerant Network Security, Domestic Network Security, Peer-to-Peer Networks Security, Security Issues in Mobile and Ubiquitous Networks, Security of GSM/GPRS/UMTS Systems, Sensor Networks Security, Vehicular Network Security, Wireless Communication Security: Bluetooth, NFC, WiFi, WiMAX, WiMedia, others

This Track will emphasize the design, implementation, management and applications of computer communications, networks and services. Topics of mostly theoretical nature are also welcome, provided there is clear practical potential in applying the results of such work.

Track B: Computer Science

Broadband wireless technologies: LTE, WiMAX, WiRAN, HSDPA, HSUPA, Resource allocation and interference management, Quality of service and scheduling methods, Capacity planning and dimensioning, Cross-layer design and Physical layer based issue, Interworking architecture and interoperability, Relay assisted and cooperative communications, Location and provisioning and mobility management, Call admission and flow/congestion control, Performance optimization, Channel capacity modeling and analysis, Middleware Issues: Event-based, publish/subscribe, and message-oriented middleware, Reconfigurable, adaptable, and reflective middleware approaches, Middleware solutions for reliability, fault tolerance, and quality-of-service, Scalability of middleware, Context-aware middleware, Autonomic and self-managing middleware, Evaluation techniques for middleware solutions, Formal methods and tools for designing, verifying, and evaluating, middleware, Software engineering techniques for middleware, Service oriented middleware, Agent-based middleware, Security middleware, Network Applications: Network-based automation, Cloud applications, Ubiquitous and pervasive applications, Collaborative applications, RFID and sensor network applications, Mobile applications, Smart home applications, Infrastructure monitoring and control applications, Remote health monitoring, GPS and location-based applications, Networked vehicles applications, Alert applications, Embeded Computer System, Advanced Control Systems, and Intelligent Control : Advanced control and measurement, computer and microprocessor-based control, signal processing, estimation and identification techniques, application specific IC's, nonlinear and adaptive control, optimal and robot control, intelligent control, evolutionary computing, and intelligent systems, instrumentation subject to critical conditions, automotive, marine and aero-space control and all other control applications, Intelligent Control System, Wiring/Wireless Sensor, Signal Control System. Sensors, Actuators and Systems Integration : Intelligent sensors and actuators, multisensor fusion, sensor array and multi-channel processing, micro/nano technology, microsensors and microactuators, instrumentation electronics, MEMS and system integration, wireless sensor, Network Sensor, Hybrid

Sensor, Distributed Sensor Networks. Signal and Image Processing : Digital signal processing theory, methods, DSP implementation, speech processing, image and multidimensional signal processing, Image analysis and processing, Image and Multimedia applications, Real-time multimedia signal processing, Computer vision, Emerging signal processing areas, Remote Sensing, Signal processing in education. Industrial Informatics: Industrial applications of neural networks, fuzzy algorithms, Neuro-Fuzzy application, bioInformatics, real-time computer control, real-time information systems, human-machine interfaces, CAD/CAM/CAT/CIM, virtual reality, industrial communications, flexible manufacturing systems, industrial automated process, Data Storage Management, Harddisk control, Supply Chain Management, Logistics applications, Power plant automation, Drives automation. Information Technology, Management of Information System : Management information systems, Information Management, Nursing information management, Information System, Information Technology and their application, Data retrieval, Data Base Management, Decision analysis methods, Information processing, Operations research, E-Business, E-Commerce, E-Government, Computer Business, Security and risk management, Medical imaging, Biotechnology, Bio-Medicine, Computer-based information systems in health care, Changing Access to Patient Information, Healthcare Management Information Technology. Communication/Computer Network, Transportation Application : On-board diagnostics, Active safety systems, Communication systems, Wireless technology, Communication application, Navigation and Guidance, Vision-based applications, Speech interface, Sensor fusion, Networking theory and technologies, Transportation information, Autonomous vehicle, Vehicle application of affective computing, Advance Computing technology and their application : Broadband and intelligent networks, Data Mining, Data fusion, Computational intelligence, Information and data security, Information indexing and retrieval, Information processing, Information systems and applications, Internet applications and performances, Knowledge based systems, Knowledge management, Software Engineering, Decision making, Mobile networks and services, Network management and services, Neural Network, Fuzzy logics, Neuro-Fuzzy, Expert approaches, Innovation Technology and Management : Innovation and product development, Emerging advances in business and its applications, Creativity in Internet management and retailing, B2B and B2C management, Electronic transceiver device for Retail Marketing Industries, Facilities planning and management, Innovative pervasive computing applications, Programming paradigms for pervasive systems, Software evolution and maintenance in pervasive systems, Middleware services and agent technologies, Adaptive, autonomic and context-aware computing, Mobile/Wireless computing systems and services in pervasive computing, Energy-efficient and green pervasive computing, Communication architectures for pervasive computing, Ad hoc networks for pervasive communications, Pervasive opportunistic communications and applications, Enabling technologies for pervasive systems (e.g., wireless BAN, PAN), Positioning and tracking technologies, Sensors and RFID in pervasive systems, Multimodal sensing and context for pervasive applications, Pervasive sensing, perception and semantic interpretation, Smart devices and intelligent environments, Trust, security and privacy issues in pervasive systems, User interfaces and interaction models, Virtual immersive communications, Wearable computers, Standards and interfaces for pervasive computing environments, Social and economic models for pervasive systems, Active and Programmable Networks, Ad Hoc & Sensor Network, Congestion and/or Flow Control, Content Distribution, Grid Networking, High-speed Network Architectures, Internet Services and Applications, Optical Networks, Mobile and Wireless Networks, Network Modeling and Simulation, Multicast, Multimedia Communications, Network Control and Management, Network Protocols, Network Performance, Network Measurement, Peer to Peer and Overlay Networks, Quality of Service and Quality of Experience, Ubiquitous Networks, Crosscutting Themes – Internet Technologies, Infrastructure, Services and Applications; Open Source Tools, Open Models and Architectures; Security, Privacy and Trust; Navigation Systems, Location Based Services; Social Networks and Online Communities; ICT Convergence, Digital Economy and Digital Divide, Neural Networks, Pattern Recognition, Computer Vision, Advanced Computing Architectures and New Programming Models, Visualization and Virtual Reality as Applied to Computational Science, Computer Architecture and Embedded Systems, Technology in Education, Theoretical Computer Science, Computing Ethics, Computing Practices & Applications

Authors are invited to submit papers through e-mail ijcsiseditor@gmail.com. Submissions must be original and should not have been published previously or be under consideration for publication while being evaluated by IJCSIS. Before submission authors should carefully read over the journal's Author Guidelines, which are located at http://sites.google.com/site/ijcsis/authors-notes .

ISSN 1947 5500
http://sites.google.com/site/ijcsis/