World Journal of Research and Review (WJRR)
ISSN:2455-3956, Volume-2, Issue-5, May 2016 Pages 50-55

On the Technical Quality Evaluation of Rapid Application Development Software Products in a Visual Environment

Laura Silvia Vargas-Pérez, Agustín Francisco Gutiérrez-Tornés, Edgardo Manuel Felipe-Riverón, Inés Zambrano-Dávila, Ricardo Peña-Galeana

***Abstract*—This paper presents a model for the technical quality evaluation of Rapid Application Development (RAD) software products in a visual environment. It addresses organizations, companies and final users that need to select in an effective and easy way the most appropriate software to develop their applications amongst those in the market. It also gives a guideline for the concrete instrumentation of the model features, such as ranking procedures. Finally, it discusses the results of the evaluation of three of these tools.**

***Index Terms*— Technical quality evaluation, Rapid Application Development tools, software product quality**

I. INTRODUCTION

Rapid Application Development (RAD) is a method used for the elaboration of software products mainly based upon the continuous interactive prototyping and design of the system with a huge involvement and participation of final users by means of computerized tools [1]. Some papers and thesis deal with this subject mainly concerned upon the quality of the method and the resulting product, but poorly about how to select the proper RAD tool from those today in the market [2]. This article presents a method for the technical evaluation and selection of the more convenient RAD software in accordance with the organization´s individual purpose.

To evaluate any software product, it is necessary first to establish its desired quality properties and then the manner of measuring them by means of a group of significant metrics [3], [4],[5], [6]. These will provide indicators, which willlead to a strategy for the technical evaluation of the product quality. It is important to do the measurements in an easy way so to interpret the results without any possible ambiguity [7], [8],[9], [13], [14].

Therefore, one must build a qualimetric model that identifies the quality components and their inter relations.Its objective is to facilitate the qualitative and quantitative evaluation of these components.The qualimetric model generally represents the entirety of the evaluation elements. Usually a tree of hierarchical structure classifies them, where the characteristics appear in the higher level, the sub characteristics in the intermediate level, and the attributes in the lowest one. In this article, we present such a model and its implementation.

Laura Silvia Vargas-Pérez, Instituto Tecnológico de Ciudad Madero, Ciudad Madero Tamaulipas, México
Agustín Francisco Gutiérrez-Tornés, Facultad de Ciencias y Tecnología de la Información, Universidad Autónoma de Guerrero, Acapulco, México
Edgardo Manuel Felipe-Riverón, Centro de Investigación en Computación, Instituto Politécnico Nacional, Ciudad de México, México
Inés Zambrano-Dávila, Facultad de Ciencias y Tecnología de la Información, Universidad Autónoma de Guerrero, Acapulco, México
Ricardo Peña-Galeana, Facultad de Ciencias y Tecnología de la Información, Universidad Autónoma de Guerrero, Acapulco, México

II. TYPES OF MEASURES

There are two types of evaluation objectives:

- To identify problems that can be rectified, and
- To compare the quality of a product with alternative products or against requirements.

This research refers to the second objective. The type of required measurements will depend on the purpose of the evaluation. If the primary purpose is to detect and to correct deficiencies, many measurements can be made within the software to visualize and to control improvements. When comparingthequalityofaproductwithalternativeproductsoraga instrequirements,itisimportanttobasethespecificationoftheeva luationonaprecisequalimetricmodel,measurementmethods and scales orrange of levels for each metric[10], [11],[12], [13].The method presented here in, allows a comparative analysis among different types of Rapid Application Development tools in a visual environment from which the user will be able to select the most appropriate to fulfill its needs[14],[15].

III. STATEOFTHEARTANDRELATEDWORKS

For some years a varied sort of quality measurement models mainly based upon international standards has been developed. These models are very useful, but they are usually very generic and so they should be adapted for their practical use. Previous works focuses on the evaluation of software development processes: Carballo [19]; Moreno and Lopez, 2004 [20]; Olsina and Covella, 2006 [21]; Piattini and Rolón, 2006 [22]; Pastor et al. 2006 [23] and others. Carballo [19] tries to estimate and control the quantitative administration of software projects. Moreno and Lopez [20] use software engineering metrics to evaluate grammatical analyzers, and focus the evaluation at the analysis process. Olsina and Covella [21] guide his efforts to the evaluation of Web application quality. Piattini and Rolón [22] deal in some of his works with the evaluation of the complexity of the business processes. Pastor et al. [23] presents a usability model to evaluate this characteristic during the analysis stage and during the software development process, within the perspective of the MDA (Model Driven Architecture). Villalba et al. present an interesting approach about to how to create qualimetric models for any particular domain [7].

In this proposal, the objective is different, since we are evaluating commercial products. Therefore, here we take as reference not only the basic elements of international standards, but also several more practical models, namely: ISO/IEC 9126 [11], ISO/IEC 14598 [12], SQUARE [16], [18], IEEE 1061 [10], MECA [13], MACS [14] and SUMI [17].

IV. METHODOLOGYANDEVALUATIONMODEL

The design of this model, bases itself on the coalition of the already mentioned. A part is adopted and adapted to conform the design of the proposed model (see Fig. 1). It is opportune to emphasize that the software products for which the technical evaluation model is designed must be already in the operational stage. Being commercial products, the information concerning their development as well as their source code are not available; thus the internal metrics are not taken into account.

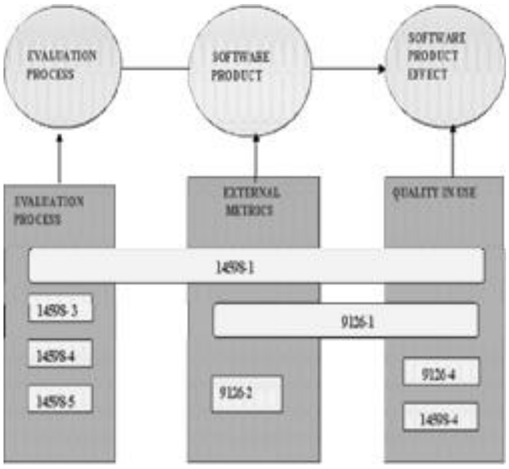


Fig. 1 Architectural model

To evaluate software quality, the user has first to determine the quality evaluation requirements. Then he specifies its design and executes the evaluation process, thus carrying out the measurements that the model includes.

This here model has six tasks (Fig. 2), each one dealing with a particular quality characteristic or property. These are subdivided into sub characteristics and then into attributes. Those attributes related to quality in use, that represent the measurement of the effect of the software product from an user point of view, will have a particular importance [6], [7], [11].

One of the purposes of this qualimetric model is to provide a range for comparison among the variety of visual environment software developing tools to any kind of user (expert or beginner). In accordance with this, it has to be a flexible one. So, the model suggested in Fig. 2 is the to be followed by experienced final users, while beginners will use the one shown in Fig. 3.

V. METRICS AND EVALUATION SCALE DEFINITIONS

To evaluate the attributes quantitative measurements are carried out by means of a given metric. The result and the so obtained value can be projected into a map on a scale. This value does not show the satisfaction level of the requirements. That is why the scale has to be divided in ranges according to different degrees of satisfaction. Some examples of how to do it are the following:

- To divide the scale into two categories: unsatisfactory and satisfactory.
- To divide these categories in five levels A, B, C and D (all them satisfactory) and E (unsatisfactory).

Level A is the best. It is the Ideal level to achieve. The product expected results would probably exceed the requirements. Level B is advisable. It considers possible to reach the expected result with the available resources. Level C is the average. The system will be performing without any malfunctioning. Level D is the lowest valid level. It is the limit for the user's acceptance since requirements will be just fulfilled. Finally, there is the level E. In this case the product does not fulfill the minimum quality requirements (see Figure).

Characteristic/ Sub Characteristic/ Attribute/ metric

1.1.1.1. Functionality/ Completeness/ Total contain/ metric
1.2.1.1. Functionality/ Consistency/ Format of components and functional elements uniformity/ metric
1.2.2.1. Functionality/ Consistency/ Processing return uniformity/ metric
1.2.3.1. Functionality/ Consistency/ Vocabulary and symbols used conventions uniformity/ metric
1.3.1.1. Functionality/ Correction/ Correct operation / metric
1.3.2.1. Functionality/ Correction/ Correct utilization of language / metric
1.3.3.1. Functionality/ Correction/ Correspondence of descriptions with objects / metric
1.4.1.1. Functionality/ Interoperability/ Components and interfaces exchange/ metric
1.4.2.1. Functionality/ Interoperability/ Data exchange/ metric
1.5.1.1. Functionality/ Standardization/ Symbols standardization/ metric
1.5.2.1. Functionality/ Standardization/ Vocabulary standardization/ metric
2.1.1.1. Reliability/ Maturity/ Time between failures/ metric
2.2.1.1. Reliability/ Recoverability/ Options to recover itself / metric
2.3.1.1. Reliability/ Tolerance of errors or failures/ Degraded processes/ metric
2.3.2.1. Reliability/ Tolerance of errors or failures/ Errors processing/ metric
3.1.1.1. Usability/ Attraction/ Attractive interaction/ metric
3.1.2.1. Usability/ Attraction/ Successful recovery/ metric
3.1.3.1. Usability/ Attraction/ Time of operation/ metric
3.2.1.1. Usability/ Diffusion/ Amplitude/ metric
3.2.2.1. Usability/ Diffusion/ Frequency of operation/ metric
3.3.1.1. Usability/ Learnability/ Demo/ metric
3.3.2.1. Usability/ Learnability / Demo efficiency/ metric
3.3.3.1. Usability/ Learnability / Tutorial / metric
3.3.4.1. Usability/ Learnability / Tutorial efficiency/ metric
3.3.5.1. Usability/ Learnability/ Documentation/ metric
3.4.1.1. Usability/ Understandability/ Adequate user interface/ metric
3.4.2.1. Usability/ Understandability/ On line aid/ metric
3.4.3.1 Usability/ Understandability/ Terminology in agreement to user / metric
3.5.1.1. Usability/ Operability/ Help utility/ metric
3.5.2.1. Usability/ Operability/ Help operability/ metric
4.1.1.1. Efficiency/ Use of time/ Efficiency in time/ metric
4.2.1.1. Efficiency/ Use of resources/ Efficiency in resources/ metric
4.3.1.1. Efficiency/ Scalability / Availability/ metric
5.1.1.1. Portability/ Installability / Installation module/ metric
5.1.2.1. Portability/ Installability/ Documentation of installation module/ metric
5.1.3.1. Portability/ Installability/ Configuration module/ metric
5.1.4.1. Portability/ Installability/ Documentation of configuration module/ metric
5.2.1.1. Portability/ Adjustability/ Independence of the hardware environment/ metric
5.2.2.1. Portability/ Adaptability/ Independence of software environment/ metric
6.1.1.1. Quality in use/ Effectiveness/ Tasks effectiveness/ metric
6.1.2.1. Quality in use/ Effectiveness/ Tasks performance/ metric
6.2.1.1. Quality in use/ Productivity/ Productive proportion/ metric
6.2.2.1. Quality in use/ Productivity/ User relative efficiency/ metric
6.3.1.1. Quality in use/ Satisfaction/ User favorite psychological effects/ metric

Fig. 2 Compacted model

1.3.1.1. Functionality/ Correction/ Correct utilization of language / metric
3.1.3.1. Functionality/ Correction/ Correspondence of descriptions with objects / metric
1.5.1.1. Functionality/ Standardization/ Vocabulary standardization/ metric
1.5.2.1. Functionality/ Standardization/ Symbols standardization/ metric
3.1.1.1. Usability/ Attraction/ Attractive interaction/ metric
3.2.1.1. Usability/ Diffusion/ Amplitude/ metric
3.2.2.1. Usability/ Diffusion/ Frequency of operation/ metric
3.3.1.1. Usability/ Learnability/ Demo/ metric
3.3.2.1. Usability/ Learnability / Demo efficiency/ metric
3.3.3.1. Usability/ Learnability / Tutorial / metric
3.3.4.1. Usability/ Learnability / Tutorial efficiency/ metric
3.3.5.1. Usability/ Learnability/ Documentation/ metric
3.4.1.1. Usability/ Understandability/ Adequate user interface/ metric
3.4.2.1. Usability/ Understandability/ On line aid/ metric
3.4.3.1. Usability/ Understandability/ Terminology in agreement to user / metric
3.5.1.1. Usability/ Operability/ Help utility/ metric
3.5.2.1. Usability/ Operability/ Help operability/ metric
5.1.1.1. Portability/ Installability/ Installation module/ metric
5.1.2.1. Portability/ Installability/ Documentation of installation module/ metric
5.1.3.1. Portability/ Installability/ Configuration module/ metric
5.1.4.1. Portability/ Installability/ Documentation of configuration module/ metric
6.3.1.1. Quality in use/ Satisfaction/ User favorite psychological effects/ metric

Fig. 3:Subset of the model suggested for the evaluation of the product by a novice user

Metric is defined as "a quantitative measure of the degree in which a system, component or process possesses a given attribute" [9]. In order to properly measure the different tool performance one must follow these guidelines:

- Observation of the software performance in order to evaluate the difference between the current execution results and the requirements specification (a view on test and quality validation).
- Unexpected occurrences on performance time or resources utilization during the software operation.

Therefore, evaluating all attributes belonging to a given sub characteristic one obtains an average value that evaluates that sub characteristic in particular. Then, evaluating all the sub characteristics of a given characteristic the user calculates another average value that evaluates that characteristic in particular. Finally, evaluating all the characteristics a new average value that corresponds to the software product as a whole is calculated. The mathematical method is the following [24]:

Quality indicator of the productt:

$$IC_t = \frac{\sum_{j=1}^{n} ICC_j}{n}$$

Where:

ICCj is the quality indicator of the characteristicj

n is the number of characteristics in the model

Quality indicator of the characteristic j:

$$ICC_j = \frac{\sum_{j=1}^{m} ICSC_k}{m}$$

Where:

ICSCk is the quality indicator of the subcharacteristic k

m is the number of sub characteristics within the characteristic k

Quality indicator of the subcharacteristic k:

$$ICSC_k = \frac{\sum_{x=1}^{k} VAA_x}{k}$$

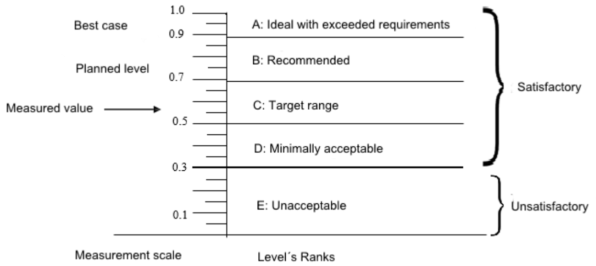
Where:

VAAx is the assigned value to the attribute x

k is the number of attributes within the sub characteristic k.

Thus, when applying the evaluation format you use three types of metrics:

- Direct instructions to the user for carrying out a specific task, taking note of certain indicators (for example: time, number of occurrences of certain event, etc.) The result will be a quantity within the proposed range (Fig. 4).
- Direct questions to the user to determine the existence of an essential attribute within the evaluated tool. The result will be an affirmative (1) or a negative (0) one (Fig. 4).
- Metrics that depend on the value of certain indicator derived from the realization of a certain task. They serve to calculate a set of parameters with values within the proposed interval (Fig 4).



Value	Fulfillment	Meaning / Interpretation	Rank
1.0	90-100	Excellent / Always	A
0.8	70-89	Satisfactory / Almost always	B
0.6	50-69	Acceptable / Regularly	C
0.4	30-49	Deficient / Sometimes	D
0	0-29	Unacceptable / Never or rare times	E

Fig. 4 Values and ranks

In order to support this model, 44 metrics were developed and documented, just as it appears in the format of Fig. 5 and 6. Another 11 metrics were adapted from SUMI [12].

Characteristic: 1. Functionality.

Sub characteristic: 1.2 Consistency.

Attribute: 1.2.3. Uniformity in the Process

Metric: 1.2.3.1 Proportion of adequate functions re-establishment from any depth level.

Method: Knowledge of functional performance.

Formula: X=1-(A/B)

A = Number of functions changed after introducing operations during a specific period.

B= Number of specific functions.

Interpretation: Stability of functional specifications objective

0<=X<=1; the closer to 1 the better

Source of reference: ISO/IEC 9126

Fig. 5 Documentation of a metric

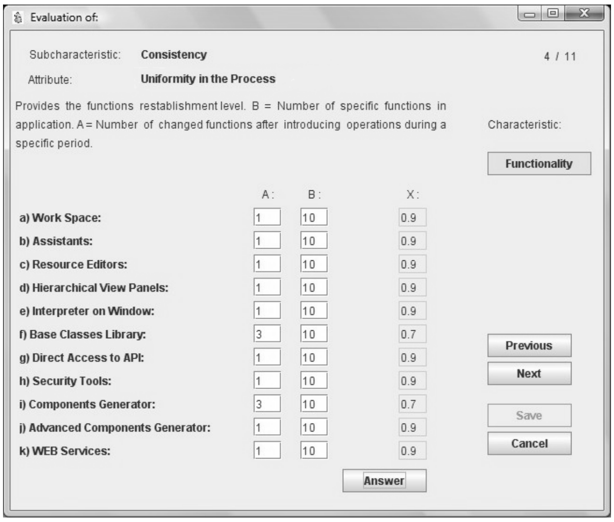


Fig. 6 Evaluation results for the Uniformity in the Process attribute

VI. PRESENTATION OF THE PROCESS AND RESULTS

To capture the evaluation data is not an easy task. For that reason, simple and comprehensive formats have been designed to facilitate the evaluation process.

Mainly the formats constitute a verification lists (checklist). These are questionnaires (or asseverations) that should be answered (or confirmed) by the user capturing one of the values corresponding to a given scale (Fig.7).

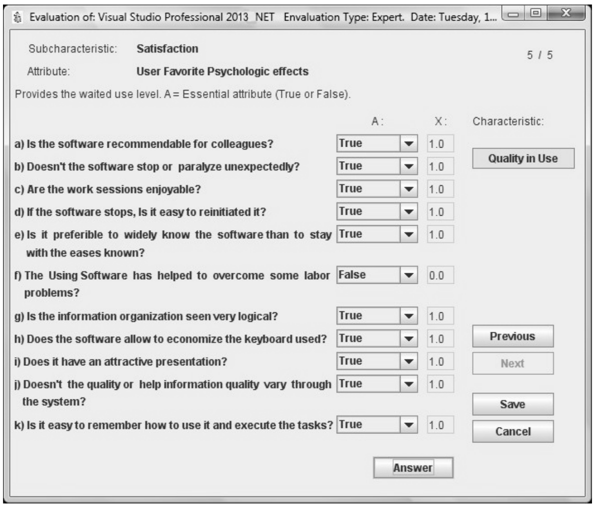


Fig. 7 Checklist example

The control matrix is a complementary tool regarding all aspects related to the supervision process and helps to plan and summarize the content and guidance of the system´s development. It usually includes a control variable (what is measured), the measurement manner, the place and moment when it is done, the standard followed, etc. Fig. 8 shows an example of part of the control matrix used to obtain the evaluation results of a particular characteristic.

At the end a final report is generated. Here the general results and percentage are captured. An outline shows the elements where the particular software product obtained a good classification quality level. Fig. 9 , 10. and 11 show the results of the evaluation of three of the still most popular RAD visual environments tools .

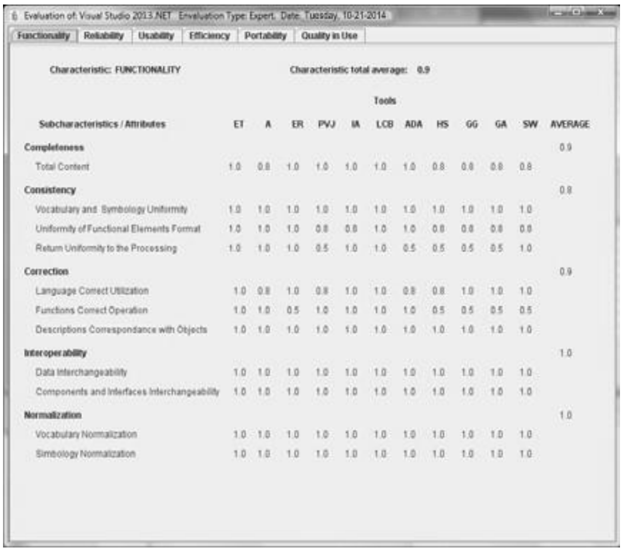


Fig. 8 Part of the control matrix for the Functionality characteristic

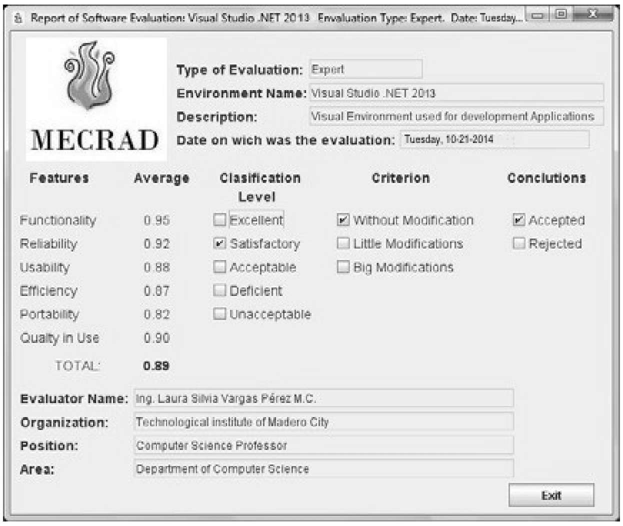


Fig. 9 Final technical evaluation report of the Visual Studio.NET environment version 2013

VII. CONCLUSIONS AND FUTURE WORKS

The results obtained through the application of the tool MECRAD [25] are the following:

The VisualStudio.Net obtained a general average evaluation of 0.89 (89%) for beginners and a punctuation of 0.88 (88%) among experts. Its weakness lies in portability. This is comprehensible, due to its dependence upon Microsoft´s Windows platform. Its quality classification level is Satisfactory, without recommendations, since it does not require modifications in its design (only updating) and therefore it is accepted thoroughly.

The results obtained from the other two products in their evaluation, have only 2% of variability. The level of quality classification obtained in these development platforms was Excellent for Net Beans and Eclipse.

To provide a more realistic assessment the final result is the combination of different users evaluation of the same type (expert or basic). This will allow a more realistic final technical report (Fig. 12).

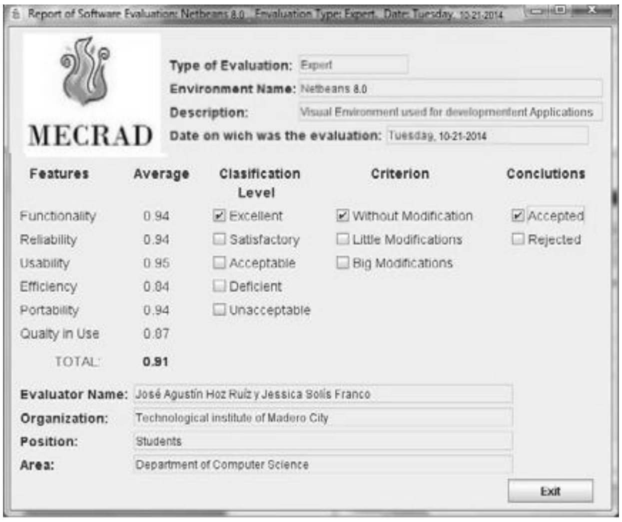


Fig. 10 Final technical evaluation report of the Net Beans visual environment version 8.0

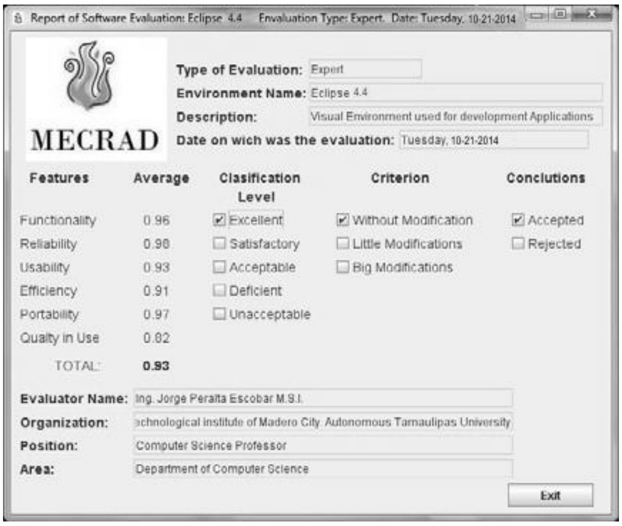


Fig. 11 Final technical evaluation report of the Eclipse visual environment version 4.4

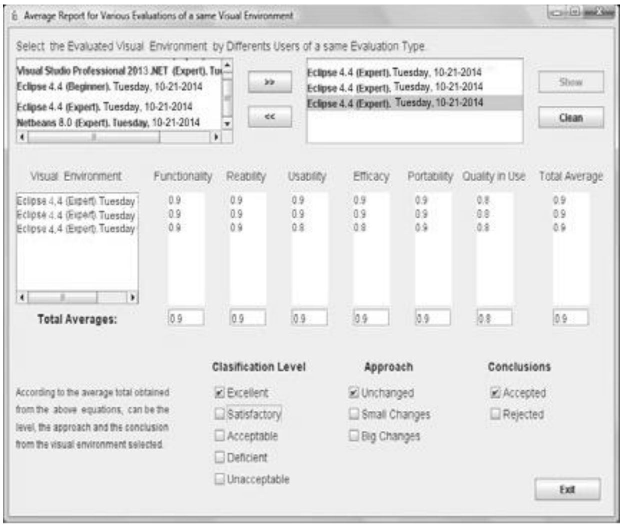


Fig. 12 Average report

Any of the three visual environment system mentioned above are considered technically advisable for application developments. For that reason, if one requires a decision about the acquisition of some of these environments, one takes in account other important parameters, such as cost, platform or environment in which the application will be developed, systems interacting within the environment and others. The model does not contemplate these parameters, since it is limited to the technical quality evaluation of the visual tools themselves.

As a future work, it would be advisable to make periodic revisions of the model for its improvement, attempting for example to introduce the evaluation of tools in the visual WEB sites environment, as well as in other kinds of environments.

REFERENCES

[1] Martin J. (1991). Rapid Application Development, Macmillan Publishing Co., Inc. Indianapolis, IN, USA. ISBN: 0-02-376775-8.

[2] Coleman G., Verbruggen R. (1998). A Quality Software Process for Rapid Application Development. Software Quality Control, Vol. 7, Issue 2, pp. 107-122.

[3] Cochea T., Silvia J. (2009). Métrica de calidad de los sistemas de información – aplicación en la certificación de calidad de un Sistema de una empresa del sector hidrocarburífero. Artículos de Tesis de Grado – ICM, Escuela Superior Politécnica del Litoral, Ecuador, http://www.dspace.espol.edu.ec/handle/123456789/4908 (Accessed last time on March 21, 2015.)

[4] León Martínez, N. E., Gómez Flórez, L.C., Pimentel Ravelo, J.I. (2011). Herramienta computacional para la gestión y evaluación de proyectos software enmarcados en actividades de Investigación, Scientia et Technica, Vol. 1. Num.47, Universidad Tecnológica de Pereira, Colombia.

[5] Villalba de Benito, M.T. (2008). Metodologías de desarrollo de modelos de calidad orientados a dominio y saplicación al dominio de los productos finales de seguridad de tecnologías de la información. Tesis de Doctorado, Departamento de Computación, Universidad de Alcalá, España. http://dspace.uah.es/dspace/bitstream/handle/10017/6569/tesis_FINAL.pdf.txt?sequence=3 (Accessed last time on March 21, 2015.)

[6] Villalba de Benito, M.T.; Fernández Sanz, L., Cuadrado Gallegos, J.J., Martinez, J.J. (2010). Software Quality evaluation for security COTS products, International Journal of Software Engineering and Knowledge Engineering, DOI: 10.1142/S0218194010004633, Vol. 20, Issue 01,p. 20-27; World Scientific Publishing (UK) ltd. http://www.worldscientific.com/doi/abs/10.1142/S0218194010004 6 (Accessed last time on March 21, 2015.)

[7] Villalba de Benito, M.T., Fernández Sanz, L., Martinez, J.J. (2010). Empirical support for the generation of domain-oriented quality models; IET Software, DOI: 10.1049/iet-sen.2009.0040, Vol. 4, Issue 1, p. 1 – 14, UK. http://digital-library.theiet.org/content/journals/10.1049/iet-sen.2009.0040 (Accessed last time on March 21, 2015.)

[8] Pressman, R. (1998). Ingeniería de Software. Un enfoque práctico, Mc.Graw Hill /Interamericana de España, S.A.U.

[9] IEEE Software Engineering Standards Collection. (1994). Standard Glossary of Software Engineering Terminology. IEEE, Std 610.12-190.

[10] IEEE Std. 1061. (1992). IEEE Standard for a Software Quality Metrics Methodology.

[11] ISO/IEC 9126. (1997). Software Quality Characteristics as Metrics. Part 1: Quality, Characteristics and Guidelines for their Use; Part 2: External Metrics. Part 3: Internal Metrics.

[12] ISO/IEC 14598. (1998). Information Technology Software Product Evaluation. (Parts 1, 2, 3, 4, 5, 6)

[13] Gutiérrez, A. (2003). Modelo de evaluación para el aseguramiento de la calidad del Software, Modelo MECA. Instituto Politécnico Nacional.

[14] Gutiérrez, A. (1999). Metodología para el aseguramiento de la calidad del Software (MACS). Instituto Politécnico Nacional.

[15] ISO and Industry Standards for User Centered Design. (October 2000). www.usability.serco.com/trump (Accessed last time on March 21, 2015.)

[16] ISO/IEC JTC C1/SC7/ WG6 N2246. (Mayo 2000). Plan y configuración de los requerimientos de calidad de software y evaluación. SQUARE2000.

[17] Human Factors Research Group. (2000). SUMI: Software Usability Measurement Inventory. European Directive on Minimum Health and Safety Requirements for Work with Display Screen Equipment (90/270/EEC). Ireland.

[18] ISO/IEC 25000:2005. (2005). Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE. http://www.iso.org/ (Accessed last time on March 21, 2015.)

[19] Carballo, R. (2004). Gestión Cuantitativa de Proyectos Software: Métricas para Estimar y Controlar. Proc. of the Ist Simposio Avances en Gestión de Proyectos y Calidad del Software. Salamanca, España. pp. 68-

[20] Moreno, M., López, V. (2004). Aplicación de las Métricas de Calidad del Software en la Evaluación Objetiva de Gramáticas Independientes de Contexto Inferidas. Proc. of the Ist Simposio Avances en Gestión de Proyectos y Calidad del Software. Salamanca, España. Oct.21-23, pp. 209-220.

[21] Olsina, L., Covella, G. (2006). Medición y Evaluación de Calidad en Uso: Un Caso de Estudio para una Aplicación E-Learning. Proc. of the 9th Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, La Plata, Argentina, pp. 317-330, ISBN-10:950-34-0360.

[22] Piattini, M., Rolón, E. (2006). Métricas para la Evaluación de Modelos de Proceso de Negocio. Proc. of the 9th Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, La Plata, Argentina, pp. 419-432, ISBN-10:950-34-0360.

[23] Pastor, O. España, S. Pederiva, I. (2006). Evaluación de la Usabilidad en un Entorno de Arquitectura Orientadas a Modelos. Proc. of the 9th Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, La Plata, Argentina, pp. 331-344, ISBN-10:950-34-0360.

[24] Gutiérrez-Tornés, Agustín F. (2001). Modelo Cualimétrico para la Evaluación de la Calidad de la Documentación del Software; Informe Técnico, CIC/IPN, No. 96, Serie: ROJA, ISBN: 970-18-6379-8; México, D.F.

[25] Vargas-Pérez, Laura S., Gutiérrez-Tornés, Agustín F. and Felipe-Riverón, Edgardo M. (2008). MECRAD: model and tool for the technical quality evaluation of software products in visual environment. Computing in the Global Information Technology, 2008. ICCGI'08. The Third International Multi-Conference on. IEEE,Athens, Greece.

Laura Silvia Vargas-Pérez. She received the B.Sc. degree in Electronic Engineering from Technological Institute of San Luis Potosí, México, in 1987, and her M Sc. degree from Center for Computing Research of the National Polytechnic Institute of Mexico in 2006. She is currently Full Professor and Senior Researcher at the Technological Institute of Ciudad Madero, Tamaulipas, Mexico. Her research interests are on different topics of Software Engineering, Software Quality, and Educative Software. Currently she is studying a PhD in Projects in Universidad Internacional Iberoamericana.

Agustin Francisco Gutierrez-Tornés.
He receivedthe B.Sc. degree in Economics from the Institute of Economics, University of Havana, Cuba, in 1970, and his Ph.D. degree from the Enterprise Improvement Research Center, Agricultural Academy (SGGW-AR), Warsaw, Poland, in 1984. Actually, he acts as Full Professor at the Faculty of Sciences and Information Technology (CyTI) of the Autonomous University of Guerrero (UAGro). His research interests are on different topics of Software Engineering.

Edgardo Manuel Felipe-Riveron.
He received the B.Sc. degree in Electrical Engineering from the Higher Polytechnic Institute Jose Antonio Echeverria, Havana, Cuba, in 1967. He received the Ph.D. degree in Technical Sciences from the Computer and Automation Research Institute, Budapest, Hungary, in 1977. He is currently Full Professor and Senior Researcher at the Center for Computing Research of the National Polytechnic Institute of Mexico. He is member of the National Researchers System of Mexico. His research areas of interest are on Image Processing and Image Analysis, Computer Vision and Pattern Recognition, in particular color quantization, retina analysis, biometric solutions, document analysis and mathematical morphology applications.

Juana Inés Zambrano-Dávila.
She received her Bachelor's degree in Mathematics at the Mathematics Academic Unit of the Autonomous University of Guerrero in 2001, and her Master's degree in Computer Sciences in 2013 from the same university. Actually, she works at this institution as Full Professor at the Faculty of Sciences and Information Technology (CyTI).Her research interests are on different topics of Software Engineering.

Ricardo Peña-Galeana.
He received his Mechanical Electricity Engineer degree in 1995, and his Master's degree in Engineering in 2004 both from the National Autonomous University of Mexico. Actually, he acts as Full Professor at the Faculty of Sciences and Information Technology (CyTI) of the Autonomous University of Guerrero (UAGro). His research interests are on different topics of Software Engineering**.**