Software Ageing Measurement Model (SAMM): An Instrument Development

Zaiha Nadiah Zainal Abidin[1], Jamaiah H. Yahaya[1] and Aziz Deraman[2]
*[1]Faculty of Technology and Information Science, Universiti Kebangsaan Malaysia,
43600 UKM Bangi, Selangor, Malaysia.*
*[2]School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu,
21030 Kuala Nerus, Terengganu Malaysia.*
zaihanadya@yahoo.com

***Abstract*—The rapid expansion in software development forced the owners and developers of a software to develop a good quality software and relevant for use in a long period of time without affecting the operation and high maintenance cost. There should be a standard measurement or indicator to monitor relevance level of the software from internal and external views of the product. Software ageing measurement is an effort to help the owners of the software to monitor the level of relevance of the application software that has been developed and operated in certain environment. This study aims to develop software ageing measurement model and therefore, the instrument for measuring the ageing should be developed as the input to identify the quality status and relevancy of the measured software. There are 3 phases in developing software ageing measurement instrument: development of measurement requirement, development of the instrument items and instrument validation. After all the processes have been implemented, the instrument is finalized and readied to be used in software ageing measurement model. In this study, the instrument was developed based on the Software Ageing Factors Hierarchical (SAFH) Framework. The measurement in the instrument uses Likert scales as the numerical values.**

***Index Terms*—Framework; Software Ageing; Software Ageing Factors Hierarchical; Software Ageing Measurement Instrument.**

I. Introduction

There are various domains that use ageing concept as a quality measurement of the product such as human [1], insulation system [2] and textile [3]. In previous studies, the researchers used the concept of ageing to understand the influential factors that led to ageing and to find solutions to resolve this issue. By studying and understanding the factors, it will be a standard rule or mechanism to overcome the issues. In software ageing, there are four dimensions that should be emphasized [4]:

i. Type of analysis in software ageing
ii. Type of system that relevance
iii. Software ageing indicator
iv. Software rejuvenation

In our study, the main focuses will be on software ageing indicator and software rejuvenation. The basic concept that should be understood to perform this study is software ageing, software measurement and software maintenance. The pioneer of software ageing [5] mentioned that there are two main things that related to software ageing; functionality of the software and how the software reacts to its environment. In previous study, most of software ageing researcher measure software ageing based on the product itself such as memory bloating, line of codes, memory leak, data corruption and file log [4-9].

In previous studies, [10-12] highlighted the external aspects of software ageing such as storage space, memory bloating and unreleased file lock. Technology is growing faster and all the internal aspect can be resolve in contrast to external factors that need to be studied more to have the solution [14]. According to [9, 10] software can be classified into two groups which are littleAging and bigAging. Based on [9, 10], further study need to be done to underline classes of software ageing and the action to be taken to ensure software stay young and relevant. In order to fulfil this requirement in the concept of software evolution, software quality, software maintenance and software ageing need to be understood and explored.

There are various studies in software evolution that touch about user needs, user satisfaction and customers' demands [14-19]. Previous researcher attempt to develop a standard measurement to measure software quality in many aspects such a process and products [23-29]. Software evolve to meet all the user requirements by making some corrective action to improve the software [18-20]. It is necessary to have software that always meet user requirement, but there are also a problem when all the flow, change and requirement not documented correctly and without any standard policy. These may lead to software performance degradation that called software ageing [4, 6, 7, 21, 22]. In this study, we focus on the development of software ageing measurement model as a standard tool for software practitioner to monitor the relevancy of their software to the user and environment. Critical study has been carried out by empirical study, discussion with experts, observation and brainstorming sessions. This paper will discuss further on how software ageing measurement instrument will be developed. It starts with introduction in the Section I. Section II presents the methodology of the proposed instrument, whilst Section III discusses on the software ageing measurement instrument. Finally, Section IV concludes this paper with a conclusion.

II. Methodology of Instrument Development

Software ageing instrument was developed by empirical study, expert discussions and brainstorming. Figure 1 illustrates the methodology used to develop the instrument. There are few steps in developing the software ageing measurement instrument as shown in the figure and will be discussed in the next following sub sections.

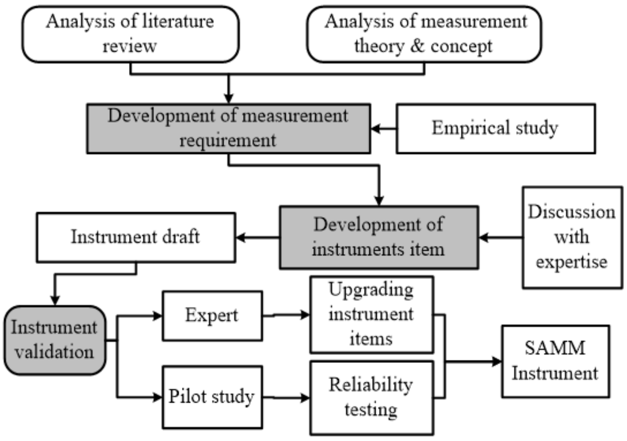
Analysis of literature review
Analysis of measurement theory & concept
Development of measurement requirement
Empirical study
Development of instruments item
Discussion with expertise
Instrument draft
Instrument validation
Expert
Upgrading instrument items
Pilot study
Reliability testing
SAMM Instrument

Figure 1: Methodology of instrument development

A. Development of measurement requirement

The instrument development process began with the development of measurement requirement that obtained from the analysis of literature review, analysis of measurement theory and concept, and findings from empirical study.

B. Development of instruments item

After all the measurement requirement has been identified, the process of developing instruments item was conducted along with the discussions and views with several experts in this area of research. The draft instrument is finally constructed and readied to be validated.

C. Instrument validation

Process of instrument validation was conducted with experts and pilot study. After instrument validation process completed, the instrument items were refined and upgraded based on the feedbacks from experts and pilot study. The reliability test was also been conducted. Finally, the completed instrument was ready to be used in software ageing measurement model.

III. Software Ageing Measurement Instrument

Software ageing factors hierarchical framework was developed as a base of software ageing measurement instrument. The hierarchical framework is shown in Figure 2. The structure of the classification framework was developed by adopting the Goal Question Metric (GQM) approach and Factor Metric Attribute Measure (FAME) method as discussed in detail in [28, 29]. In this framework, 5 elements have been recognised to be executed sequentially in Software Ageing Factors Hierarchical Framework (SAFH). The elements are goal, factor, construct, item and measurement.

A. Element 1: Purpose/Goal

In the first element, the goal to be achieved is defined precisely. In this study the goal to be achieved is software ageing measurement. It is important that the goal is defined clearly so that the next elements which are factors will be constructed correctly. All the factors will reflect the final goal of this task.

B. Elements 2: Factors

The second level of this framework is factor. It has to be identified in the plot associated with the goal. Previous study [had revealed that there are four factors that have been identified associated and influenced the software ageing which are functional, human, product profile and environment. Functional factor relates to the usability of the software. Software that cannot function according to user specification then it is considered as ageing. Human factor relates to people in terms of management, users, education, experience, knowledge and popularity. When people do not want to use this software anymore then it is considered to be in the phase of ageing. While for product profile, the aspects that need to be taken into account in this regard are date of acquisition, purchase, production, technology and software life cycle. The forth factor in this framework is environment that is considered as an external factor which involves accessories, alternatives and technological changes.

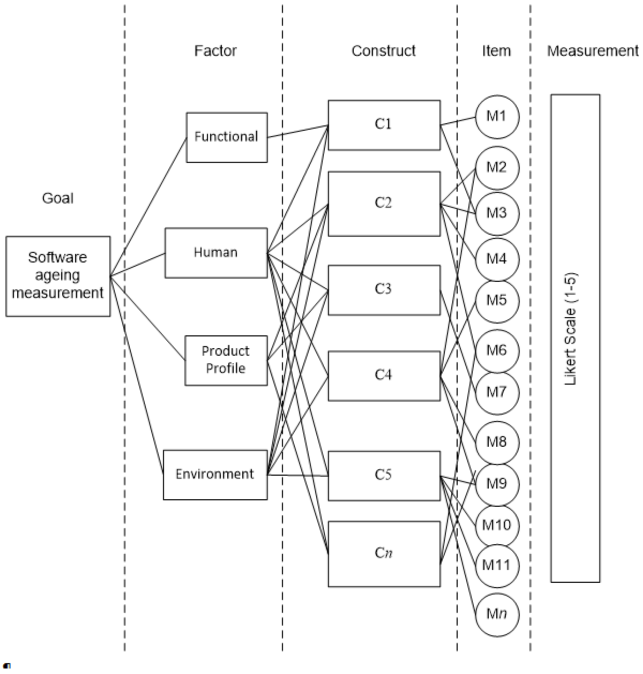

Figure 2: Software Ageing Factors Hierarchical Framework (SAFH)

C. Element 3: Constructs

Element 3 in this hierarchy is construct. It is plotted from four main factors defined earlier which are human, functional, product profile and environment. In this instrument, constructs that derive from the factors are altogether contain 23 classes which include Adaptability, Stability, Performance, Interactivity, Popularity, Knowledge, Experience, Training, Satisfaction, Support system, Adaptability, Technology suitability, Training content, Software satisfaction, Rationality, Maintenance support, Policy & Documentation, Environment adaptability, Environment change stability, and Technology acceptance. Each of the 23 constructs will be detailed and dispersed into items.

D. Element 4: Items

Every construct has its own items that has been classified thoroughly through empirical study, expert review and brainstorming sessions. This item will help researcher to find more reliable result on software ageing. The items are considered the most measureable metrics that can help users to evaluate or measure a specific software based on ageing phenomenon.

E. Elements 5: Measurements

The next element is measurement. Every item in the instruments are measured using Likert scales 1 to 5. Scales 1 to 5 representing the rate from highest to lowest. As an example, in a human factor there is a training construct (sub factor) and in the training construct there is an item (question) “Training is needed before using the software:” and the answer is in a Likert scale which are 1 = Strongly agree, 2 = Agreed, 3 = Simple, 4 = Disagree and 5 = Strongly disagree.

Based on Software Ageing Factors Hierarchical (SAFH) Framework, an instrument to measure the software ageing was developed. In this instrument, there are four factors that we used from result of empirical study, expert review, brain storming and literature review. From four factors, we derive 23 construct as shown in Table 1. From the construct we identify a suitable item to be used in the instrument of software ageing measurement.

Table 1
Instrument Construct

Factor	Construct	No of item	Total item	Average Alpha Cronbach
Functional	Adaptability	3	15	0.955
	Stability	4		
	Performance	2		
	Interactivity	6		
Human	Popularity	4	25	0.962
	Knowledge	2		
	Experience	2		
	Training	7		
	Satisfaction	7		
	Support system	3		
Product Profile	Adaptability	2	36	0.957
	Stability	4		
	Technology suitability	1		
	Training content	2		
	Software satisfaction	11		
	Rationality	3		
	Maintenance support	5		
	Policy & Documentation	4		
	Popularity	4		
Environment	Environment adaptability	1	6	0.971
	Environment change stability	3		
	Technology acceptance	1		
	Popularity	1		

In functional factors there are four constructs which are adaptability, stability, performance and interactivity. Total item that derived in functional factor are 15 items. In human factors there are six constructs which are popularity, knowledge, experience, training, satisfaction and support system. Total item that derived in human factor is 25 items as shown in Table 1. In product profile factors, there are nine constructs including adaptability, stability, technology suitability, training content, software satisfaction, rationality, maintenance support, policy & documentation and popularity. There are 36 items that derived from this factor. The forth factor is environment that contains 4 construct and 6 items. The constructs are environment adaptability, environment change stability, technology acceptance and popularity. There are 6 items that derive from environment factor.

The reliability test was conducted on this developed instrument. Table 1 shows the result. It shows that the average of Alpha Cronbach is above 0.900 which means the reliability testing result is good and all the items can be used in this instrument.

IV. Conclusion

Software need to be monitored thoroughly to maintain its quality and relevancy to the user. In this dynamic operating environment of software, software changes very fast and the relevancy of the software needs to be measured and maintained. In order to maintain its quality and relevancy, a standard mechanism needs to be established and followed. As a solution, this study proposes a standard measurement model as guidance to software owner to monitor the performance and ageing progress of the software. The development of Software Ageing Measurement instrument has been discussed in this paper and will be used as an input in the software ageing measurement model. In the next future work, formulation of measurement and algorithm will be developed to compute the relevance result as an indicator of software ageing index. The development of software ageing measurement model will be discussed in future.

Acknowledgment

This research project was funded partly by the Fundamental Research Grant Scheme, Malaysia Ministry of Higher Education which was granted under Universiti Kebangsaan Malaysia (FRGS/1/2012/SG05/UKM/02/10) and the Universiti Malaysia Terengganu research grant under FRGS vot 59297.

References

[1] A. R. Torres, and M. S Kiirzner, “Ageing and longevity are related to growth hormone/insulin-like growth factor-1 secretion,” *International Journal of Experimental, Clinical, Behavioural and Technological Gerontology*, vol. 48, no. 6, pp. 401-407. 2002.

[2] A. C. Gjerde, “Multi factor ageing models - origin and similarities,” in *IEEE Electrical Insulation Magazine*, vol. 13, no. 1, pp. 199-204. 1997.

[3] R. B. Bresee, “General effects on ageing textile,” *Journal of The American Institute of Conservation.*, vol. 25, no. 1, pp. 39-48, 1986.

[4] D. Cotroneo, R. Natella, R. Pietrantuono, and S. Russo, “Software aging and rejuvenation: Where we are and where we are going,” in *2011 IEEE Third Int. Work. Software Aging Rejuvenation*, 2011, pp. 1-6.

[5] D. L. Parnas, “Software aging invited,” in *Proceedings of 16th International Conference on Software Engineering.*, 1994, pp. 279-287.

[6] Y. Huang, C. Kintala, N. Kolettis, and N. D. Fulton, “Software rejuvenation: analysis, module and applications,” in *Twenty-Fifth International Symposium on Fault-Tolerant Computing. Digest of Papers*, 1995, pp. 381-390.

[7] M. Grottke, R. Matias Jr. and K. S. Trivedi, “The Fundamentals of Software Aging,” in *Proceedings of the International Workshop on Software Aging and Rejuvenation*, 2008, pp. 1-6.

[8] D. Cotroneo, R. Natella and R. Pietrantuono, “Is software aging related to software metrics?,” in *2010 IEEE Second International Workshop on Software Aging and Rejuvenation*, San Jose CA, USA, 2010, pp. 1-6.

[9] D. Cotroneo, R. Natella, R. Pietrantuono, and S. Russo, “Software aging analysis of the Linux operating system,” in *2010 IEEE 21st International Symposium on Software Reliability Engineering*, 2010, pp. 71-80.

[10] B. Wah, "Software aging and rejuvenation," in *Wiley Encyclopedia of Computer Science and Engineering*. John Wiley & Son, Inc, 2008.

[11] T. Thein, "Proactive software rejuvenation solution for software aging," 2011, Available at http://eurosoutheastasia-ict.org/wp-content/plugins/alcyonis-event-agenda//files/Thandar-Thein.pdf. [Retrieved 9/2/2012]

[12] A. Deraman, *Memburu Kualiti Perisian* (Inaugural speech). UKM Publisher, 2011.

[13] M. M. Lehman, "Programming systems growth dynamics," in *Info-Tech State of the Art Lectures*, no. 18, pp. 391-412, 1974.

[14] L. A. Belady and M.M. Lehman, "A Model of Large Program Development," *IBM Systems Journal*, vol. 15, no. 1, pp. 225-252, 1976.

[15] M. M. Lehman, *Program Evolution: Processes of Software*. San Diego, CA, USA: Academic Press Professional, Inc., 1985, pp. 538.

[16] V.T. Rajlich & K.H. Bennet, "A Staged Model for the Software Life Cycle," *Journal Computer*, vol. 33, no. 7, pp. 66-71 ,2000.

[17] G. Xie, J. Chen, and I. Neamtiu, "Towards a better understanding of software evolution: An empirical study on open source software," in *2009 IEEE International Conference on Software Maintenance*, 2009, pp. 51-60.

[18] I. Sommerville, *Software Engineering*. 9th Edition, Addison Wesley, 2010

[19] A. April, J. M. Desharnais and R. Dumke, "A formalism of ontology to support a software maintenance knowledge-based system," in *The Eighteenth International Conference on Software Engineering Knowledge Engineering Conference (SEKE06)*, 2006, pp. 331–336.

[20] D. Cotroneo, R. Natella, R. Pietrantuono, and S. Russo, "Software aging and rejuvenation: Where we are and where we are going," in *2011 IEEE Third Int. Work. Software Aging Rejuvenation*, 2011, no. 30, pp. 1–6.

[21] J. H. Yahaya, Z. N. Zainal Abidin, N. M. Ali and A. Deraman, "Software ageing measurement and classification using goal question metric (GQM) approach", in *Proceeding of 2013 Science and Information Conference*, 2013, London, UK, pp. 160-165.

[22] I. Tervonen, "Support for quality-based design and inspection," *IEEE Software*, vol. 13, no. 1, pp. 44–54, 1996.

[23] J. H. Yahaya, *The Development of Software Certification Model Based on Product Quality Approach*. PhD Thesis, UKM, 2007.

[24] J. H. Yahaya, A. Deraman, and A. R. Hamdan, "Software certification model based on product quality approach," *Journal of Sustainability Science and Management*, vol. 3, no. 2, pp. 14-29, 2008.

[25] J. H. Yahaya, A. Deraman and A. R. Hamdan, "Continuously ensuring quality through software product certification: A case study," in *2010 International Conference on Information Society*, 2010, pp. 183-188.

[26] N. Drouin, M. Badri and F. Touré, "Analyzing software quality evolution using metrics: An empirical study on open source software," *Journal of Software*, vol. 8, no. 10, pp 2462-2473, 2013.

[27] P. Lew, "Software Quality Metrics, Do and Dont's," in *Software Quality Improvements*, 2014.

[28] Z. N. Zainal Abidin, J. H. Yahaya and A. Deraman, " Kerangka hirarki faktor penuaan perisian: satu kajian empirik di kalangan pengamal perisian di Malaysia," *Jurnal Teknologi*, vol. 78, no. 8, pp. 139-151, 2016.

[29] Z. N. Zainal Abidin, J. H. Yahaya and A. Deraman, " Software Ageing Measurement Model (SAMM): A conceptual model," in *2015 International Conference on Electrical Engineering and Informatics (ICEEI)*, 2016, pp. 456-461.