The Implementation of Software Anti-Ageing Model towards Green and Sustainable Products

Zuriani Hayati Abdullah[1], Jamaiah Yahaya[2], Siti Rohana Ahmad Ibrahim[3], Sazrol Fadzli[4]
Faculty of Information Science and Technology,
Universiti Kebangsaan Malaysia Bangi, Selangor, Malaysia

Aziz Deraman[5]
School of Informatics & Applied Mathematics,
Universiti Malaysia Terengganu, Kuala Terengganu,
Malaysia

***Abstract*—Software ageing is a phenomenon that normally occurs in a long running software. Progressive degradation of software performance is a symptom that shows software is getting aged and old. Researchers believe that the ageing phenomenon can be delayed by applying anti-ageing techniques towards the software or also known as software rejuvenation. Software ageing factors are classified into two categories: internal and external factors. This study focuses on external factors of software ageing, and are categorized into three main factors: environment, human and functional. These three factors were derived from empirical study that been conducted involving fifty software practitioners in Malaysia. The anti-ageing model (SEANA model) is proposed to support in preventing the software from prematurely aged, thus prolong its usage and sustainable in their environment. SEANA model is implemented in collaboration with a government agency in Malaysia to verify and validate the model in real environment. The prototype of SEANA model was developed and applied in the real case study. Furthermore, the anti-ageing guideline and actions are suggested for ageing factors to delay the ageing phenomenon in application software and further support the greenness and sustainability of software products.**

***Keywords*—Software ageing factor; ageing prevention; software anti-ageing model; SEANA model; SeRIS Prototype System; Green And Sustainable Product; Emprirical Study**

I. INTRODUCTION

As the increase of dynamic software requirements nowadays from users and stakeholders, software development process is becoming more complex and resulting the degradation of software performance and software quality [1][2]. If this happens to the software which is operating in certain environments, it may get aged prematurely and no longer relevant in their environment. Users may refuse to use the software anymore because it does not fulfil and satisfied the requirements and expectation. Progressive degradation of software performance, such as software crash or hang and accumulation of software errors are reported as the phenomenon of software ageing. The ageing of the software is caused by two factors which are by the changes that have been made throughout its execution and also cause by the failure to adapt with the dynamic environment [3].

Software ageing may occur when there are accumulation of errors or software failure throughout its execution. However, it does not affect or change the functionality of a software, but its effects on the time responsiveness of the software and user satisfaction over the software [4]. Software failure is closely related to the software downturn during its life cycle. The problem led to a progressive decline in software performance, and caused the software to not function properly. This degradation process is called software ageing [5][23]. Previous studies revealed that we could slowed down software ageing by identifying the influential factors of the ageing phenomenon. There are two types of ageing influential factors which are internal and external factors. However, there are very few studies focuses on external factors [4] [8] that are closely related to software quality in application software. Based on our initial investigation has discovered that some applications get old and aged as early as three years and thus forced the users to not used the application anymore. In this scenario, the application ages prematurely. Currently there is no software anti-ageing mechanism or guideline to assist users or developers to measure and guarantee the software still relevant and young in their operating environment. However, in general, sustainability in software is associated with the ability to operate in a longer time and viable in their environment.

This paper discusses the research background and related works in Section II, the empirical study conducted in this research is presented in Section III. Section IV of this paper presents the development of the anti-ageing model while the validation and the implementation of the model are discussed in Section V. The anti-ageing actions are introduced and presented in Section VI, and lastly concludes with a discussion and conclusion.

II. BACKGROUND STUDY AND RELATED WORKS

The phenomenon of ageing is applicable in software which is operating in certain environments. By identifying the significant factors and causes of software ageing, it can ageing effect to environment, organization delay and help in preventing the occurrence of ageing phenomenon. This process may be stated as anti-ageing or a rejuvenation process of a software product. Currently in software engineering, the rejuvenation process of a software is considered as one of the mechanism for handling faults tolerant or failure in the long running software [6][21]. Hence, it is essential to find solutions on how to prevent or slow down the ageing process in order to maintain the relevancy of the software and still meet their business requirement. Next section will discuss software ageing issues and software and economics.

This work is supported in part by the Arus Perdana Grant of UKM (AP-2017-005/3).

A. Software Ageing Issues and Effects

Software ageing has been studied by several researchers as early as in 1990s. Previous studies indicated that issues and challenges related to software ageing in computer software area are normally associated with accumulation of undetermined threads or failure, data corruption, memory related problem such as memory leaking and bloating, data-files fragmentation, residual defects, unreleased files lock, memory lack and overruns [7][8].

Nowadays, software ageing does not only associate to computer software or system software but also being investigated that relates to mobile application such as android and windows mobile application [9][10][11]. Mobile was running for a longer time without rebooted or shut down compared to computer. Therefore, software ageing in mobile devices lead to an extensive challenge to ultimate the user experience and satisfaction. Software ageing issues are not limited to mobile and computer software only but now extended to the ageing phenomenon in cloud computing environment [5][12]. This shows that software ageing is a relevant issue to be explored and investigated further.

Good quality software will delay in the occurrence of ageing and prolong their usage and relevancy. This is supported by the previous researchers who believed that by maintaining good quality of software, it can somehow prevent or minimize the error or failure and thus results in user satisfaction when using the software [13]. Good quality software is referred to the technical behaviour of the software and end user's perspective towards the software, which measures the satisfaction and fulfil expectation of the software. The study done by Yahaya et al. [13] reveals that with these two quality characteristics and measurements will maintain and ensure the software are relevant more longer of time in their operating environment.

In addition to good quality software, Matias, Trivedi and Maciel [14] claimed that software maintenance must be implemented systematically to ensure the optimum quality throughout software life cycle. In software engineering, there are four main maintenance activities which are corrective, adaptive, preventive and perfective. These activities may control and delay the ageing progress of software. For instance, preventive maintenance can help to slow down the occurrence of failures determinable to this cause.

Previous studies revealed that software ageing might gave negatives effects or influences in various aspects [2][4]. For examples its gives drawback to organizational level. Ageing effects on the operating system at resource level such as non-released memory, round-off and data file fragmentations and also debug errors. These delay the work and schedule because of slow time responsiveness on running application.

B. Software Anti-Ageing and Rejuvenation

Software ageing is irresistible manifestation, but there are few studies on how to deal with ageing phenomenon in software. As mentioned earlier, software ageing can be delayed by adopting two approaches which are through software anti-ageing and software rejuvenation. There is a difference between these two approaches where software rejuvenation is used once software is detected ageing, while anti-ageing software can be applied before software becomes old in order to delay the ageing process [15].

According to [12], software rejuvenation process is not a complex task but a very effective technique in increasing the availability of a software by rebooting and refreshing the software. The rejuvenation technique is used to revive ageing software after the ageing status is detected [16][24]. Cotroneo et al. [17] suggests that software maintenance activities are considered as the anti-ageing process where these activities are used to delay the ageing of software. There are four maintenance activities that can be used as software anti-ageing techniques, which are adaptive, corrective, perfective and preventive maintenance [14]. Adaptive maintenance refers to adapting to a new environment or sometimes refers to adapting to new requirements. Corrective maintenance refers to maintenance to repair fault and perfective maintenance refers to perfecting the software by implementing new requirements. In other case it refers to maintaining the functionality of the system, improving its structure and its performance. While preventive maintenance involves performing activities to prevent the occurrence of errors. It tends to reduce the software complexity thereby improving program understandability and increasing software maintainability. In this maintenance activity comprises documentation updating, code optimization, and code restructuring. This is also known as software re-engineering.

It is crucial to prevent software ageing because it not only effects software system, but also effects user and the universe in general. There was a fatal incident that happened about twenty-six years ago where twenty eight soldiers dead and hundreds people were injured because of software failed to detect an Iraqi Scud missile and it strucked the American army barracks [18]. Based on various issues on software ageing discussed in this paper has motivated and led us to explore more on software ageing phenomenon.

C. Green and Sustainable Software Product

Green software as part of information technology (IT) covers environmental sustainability, economic energy efficiency and total cost of ownership, which includes the cost of disposal and recycling. It also refers to the application of IT to create the energy efficient and environment to maintain successful business processes and practices [25]. It incorporates three dimension that are greening IT systems and usage, using IT to support environmental sustainability and also using IT to create green awareness in a way to improve environmental sustainability [26].

Green and sustainable software can be defined as a software with direct or indirect negative impact on economy, society, human being and environment that result from development, deployment and usage of the software. It should have minimal and positive effect towards sustainable development [27]. In addition, Erdelyi [28] defines green software as the development and operation of the software that produce minimal disposal and waste as possible [28].

III. Empirical Study

This section disscusses on the empirical study that was conducted in Malaysia. The empirical study was conducted through a survey to identify the awareness and acceptance of the software ageing issues and concerns among software practitioners. Second objective of the the survey was to examine and classify the ageing factors identified from literature study that influence the ageing of software. The survey was carried out involving respondents from agencies in public and private sectors. The respondents were chosen using purposive sampling in the group of software practitioners in Malaysia. The background of the respondent may came from diverse types of organization background as shown in Table I.

Questionnaires were distributed by two methods physical questionnaire and online questionnaire (through Google form). Fifty respondents have participated and responded to this survey. Table I shows the respondents's background and their organisations. Majority of the respondent are from service, public administator and ministry department (42%). Respondents are also came from various other backgrounds such as Computer/Security System (8%), Computer Engineering/Design (6%), Software Development (2%), Training/Education/ Consultancy(6%), Internet based organization (20%), Internet based Business e-commerce/web hosting (4%), Healthcare (2%), Intergration system (2%) and others (10%). Fig. 1 illustrates that most of the respondents (48%) have three to ten years working experience, 46% of the respondents have working experience less than three years, while only 6% have working experience from eleven to more than twenty years.

TABLE I. Background of Organisation Function

	Background of Organisation/ Function	Percentage (%)
1	Service/Public Administrator/Ministry Dept.	42%
2	Software Development	2%
3	Computer/Security System	8%
4	Computer Engineering/Design	6%
5	Telecommunicating/Networking	2%
6	Internet based Business (ASP)/ e-commerce/ web hosting	20%
7	Healthcare	2%
8	Integration System	2%
9	Training/Education/ Consultancy	6%
10	Others	10%
Total :	100%	

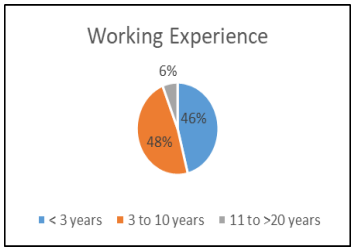

Fig 1. Service Period.

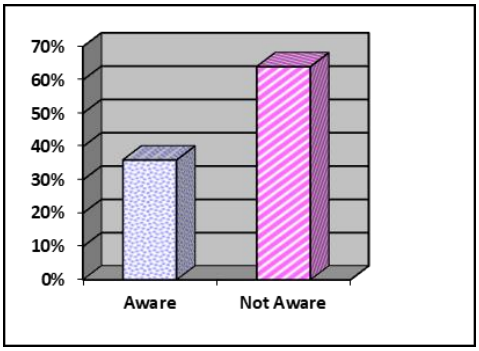

Fig 2. Software Ageing Awareness.

TABLE II. Software Ageing Factors

Factor	Metrics	Mean	%
Functionality	Time responsiveness	3.62	72.4
	Software are unable to meet the user's needs	3.6	72.0
	Failure to function as user's intended	3.58	71.6
	Progressive performance degradation	3.52	70.4
	Software is no longer relevant	3.52	70.4
	High frequency of software error	3.4	68.0
	Failure to upgrading the functionality	3.34	66.8
	Failure to get support	3.32	66.4
	User Interface (UI)	2.98	59.6
Environment	Dynamic to environment changes	3.42	68.4
	Lack of cost for software maintenance	3.42	68.4
	Software not compatible with current hardware technology	3.4	68.0
	Hardware changes	3.36	67.2
	Business process changes	3.22	64.4
	Business need	3.12	62.4
	Changes of software technology	3.06	61.2
Human	Lack of expertise in upgrading and maintaining software	3.4	68.0
	Weak software quality practices among practitioners	3.32	66.4
	Inefficient software management by management team	3.14	62.8
	Software is not user friendly	3.08	61.6

Based on the first objective of the empirical study, the analysis shows that most of the respondents (64%) are not aware and realise the presence of the phenomenon of software ageing in their operational software. Nevertheless, they agree that they have experienced in the scenario of ageing occurrence in their daily software operation (refer to Fig. 2). Majority of respondents reveal that there are no standard mechanism or policy that can be referred to measure the quality states of the software. This finding discovers the inadequate of awareness among software practitioners and software developers in software quality and related aspects. Inadequate of awareness in software quality practices can be one of possible factors that influence to the occurrence of software ageing.

This study also determines other possible factors that might influence and contribute to the software ageing phenomenon for application software. The initial discover and indentification of metrics were done through literature study and brainstorming approach among research team and experts. They include software engineering experts, software engineers and academicians in Malaysia. Twenty initial metrics have been identified and verifies through this empirical study. Table II shows the findings which verified the ageing factors. From the findings, the software ageing factors are further mapped and classified into three categories which are functionality, environment, and human.

The percentages shown in Table II are obtained from the score given by the respondents of this survey. The higher percentage means that the factor has high influence and crucial toward the ageing process. The study reveals that functionality factor contributes the highest percentage (76.4%), environment (65.72%) and human factor (64.7%) (refer Table II). This indicates the importance of these factors based on respondents perceive and perspective toward the association of ageing occurrence and phenomenon in software environment. Time responsiveness shows the highest score proving that slow response of the software will contribute to the ageing of application software. Software ageing may result in slowing the system performance in performing tasks and therefore, contributes to user dissatisfaction towards the software. Based on respondent's feedback, user interface metric obtains the lowest percentage which is 59.6%. However, software that have dull and unattractive user interface and not user friendly can also lead to software ageing.

IV. SEANA: THE SOFTWARE ANTI-AGEING MODEL

The development of anti-ageing model (or SEANA model) is based on the findings from the previous empirical study discussed in previous section as well as literature findings. The SEANA model comprises of the ageing factors and metrics (the ageing instrument), software ageing assessment process and reporting process. The last two components include the measurement algorithm, ageing levels, anti-ageing guideline and actions. SEANA model is demonstrated in Fig. 3. The following sub sections explain each of the components in this model.

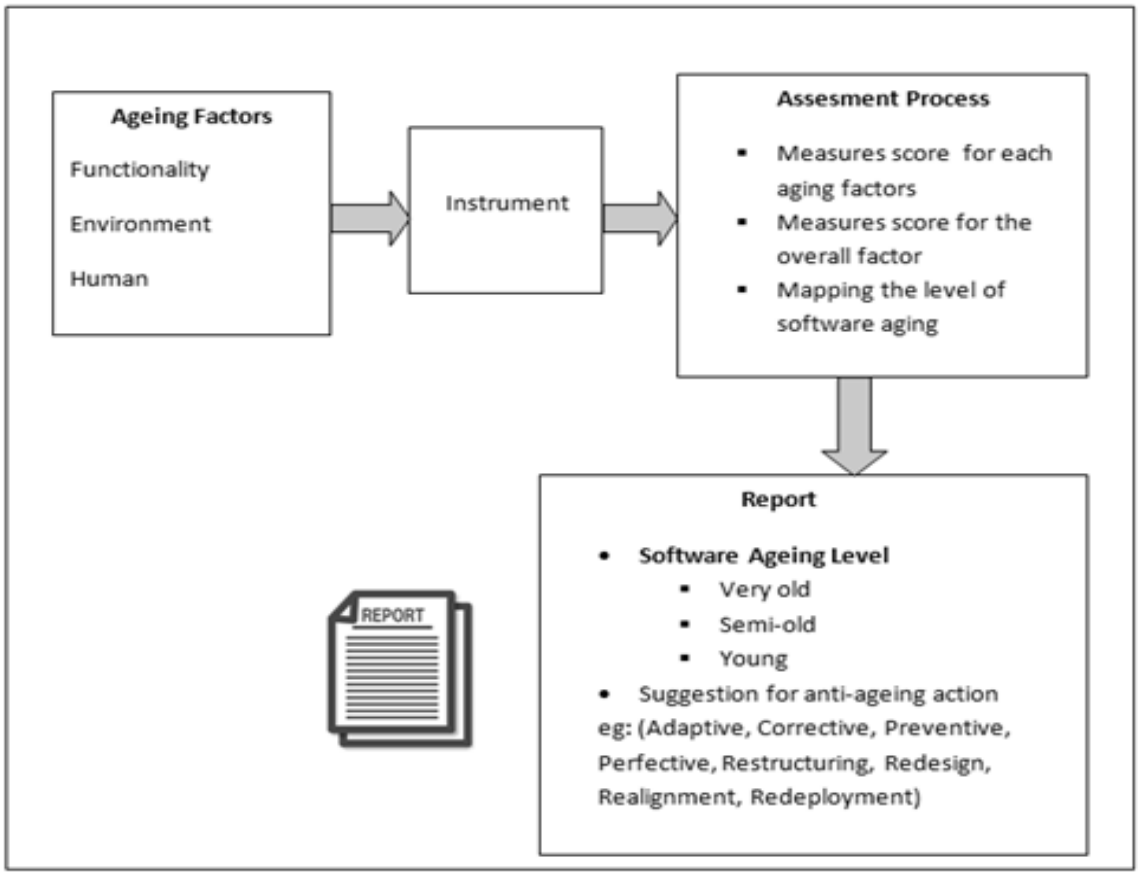

Fig 3. Software Anti-Ageing Model (SEANA).

A. Ageing Factor

The ageing factors comprises in SEANA model are designed to be used and applied by different users. Therefore, a proper and systematic instrument is needed. The objective of the instrument is to measure the ageing level of the targeted software. It can be used by the organisation or company, the stakeholder or the owner of the application software who wants to investigate the ageing status of the software. The instrument is designed in three categories of software ageing factors (which are functional, environment and human) and twenty metrics that have been identified and verified in the empirical study discussed in previous section.

All metrics in the instrument are formulated and designed to be answered by the respondents by giving scores between 1 to 4 (Likert scale of four).

B. Assessment Process

The second component of this model is the process of assessment. The assessment process component aims to do the following tasks:

- To measure the score for each categories of ageing factors that have been indicated in the instrument.
- To measure the ageing score of the overall factors.
- To map the score obtained in the assessment with the ageing level.

C. Assessment Report

The third component is the report. This component is the report generated after the completion of the assessment process. The report consists of the ageing index level, which are defined as young, semi-old and very old. If the assessment result shows that the ageing score is very old, the suggestion of anti-ageing action will be included in the report. The anti-ageing action will be suggested to the tested software according to highest scores of ageing factors obtained by the respondent during software assessment. Table III shows the ageing classification level based on the score obtained in the assessment. The ageing levels or classifications are adopted from [19] by defining ageing level as big ageing and little ageing. However, for this study purposes and compatibility, we classify software ageing into three level which are young, semi-old and very old.

TABLE III. CLASSIFICATION OF SOFTWARE AGEING

Class	Score (%)
Young	68- 100
Semi-old	35 - 67
Very Old	1-34

V. VALIDATION AND IMPLEMENTATION IN CASE STUDY

This section discussed on the validation and implementation of SEANA model in real case study.

A. The Case Study

The case study has been conducted in order to validate and implement the proposed model. We conduct a case study in one of the semi-government agency in east coast of Malaysia. We choose this agency (referred as KET) because they develop their own application in-house and have their own maintenance team to monitor their system application state. The aim of this case study was to assess three application systems that operated in their environment (referred as App1, App2 and App3 System).

1) App1 system: App1 System is an information system that manages the geographical data for a particular region. The system function similar to Google Maps but only stores and keeps the data in the east coast area of Malaysia in order to assist KET to find the rural area for ITC development purposes. Table IV shows the result obtained from the case study for App1 system.

TABLE IV. ASSESSMENT RESULTS FOR APP1 SYSTEM

Factors	Average Score (1-4)	Percentage
Functionality	1.78	44.5%
Environment	1.86	46.5%
Human	2.00	50.0%

Based on the result, the average cumulative score for this software product is 47% which is mapped into the ageing index level of Semi-old. A post assessment meeting, and review with the owner of the system discovered that the App1 System performed normally slow to retrieve geographical information such as images or maps. However, the functionality of the software is still in good condition because they practice the software maintenance activities and continuously upgrading the system regularly in order to achieve the user satisfaction.

2) App2 system: The second selected system to be tested in this case study was App2 System. It is a document management system that allows KET staff to manage document online such as letters, memos and filing. Table V shows the result obtained from this assessment based on the three factors.

TABLE V. ASSESSMENT RESULTS FOR APP2 SYSTEM

Factor	Average Score (1-4)	Percentage
Functional	2.56	64%
Environment	2.71	67.8%
Human	2.75	68.8%

3) App3 system: The third assessment of application software is App3 system. App3 System helps KET to monitor and manage the development of Rural Transformation Centre (RTC). The result of this assessment is shown in Table VI.

TABLE VI. ASSESSMENT RESULTS FOR APP3 SYSTEM

Factor	Average Score (1-4)	Percentage
Functional	3.22	80.5%
Environment	3.00	75.0%
Human	3.00	75.0%

Based on the computed result, App3 System ageing index is 76.8% which is equivalent to Young in ageing index level. We conducted a review and meeting session to verify the scored obtained for the system with the system owner and the owner agreed with the result. They claimed that App3 System is still in excellent condition and has been used actively by KET staff. Feedback from system owner is consistent with the finding that we gained from the case study.

Based on the result obtained for App2 shows that App2 System scored is 66.9% which is mapped into ageing index level of Semi-old. Feedback from the software owner claims that App2 System does not have any major problem but the application system has difficulties to upgrade some of the new functions that require by the staff.

B. *The Prototype of SEANA Model*

This section presents the prototype which developed in this research in order to validate and automate the ageing process as defined in SEANA model. The prototype is called the Software Anti-Ageing and Rejuvenation Index System (or SeRIS). The development of SeRIS was based on prototyping approach. The system was undergone through alpha and beta testing to validate the correctness and verify based on actual user's requirements. It was also being validated and applied in specific software product in real environment. This confirmation study was carried out collaboratively with industry. Feedbacks from the testing and validation activity were used in refining the SEANA model and SeRIS prototype system.

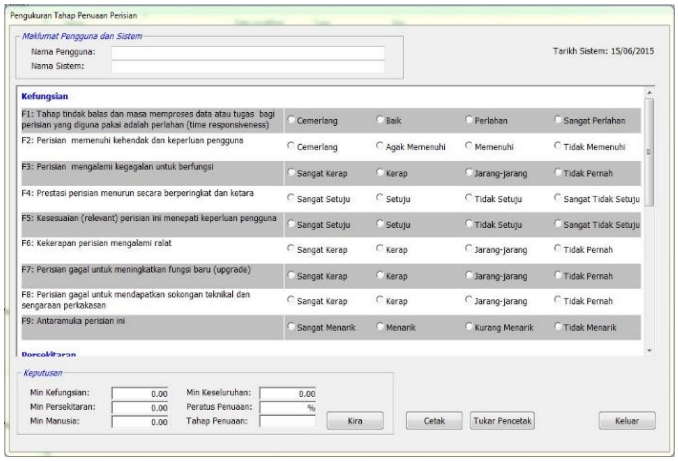

Fig 4. SeRIS Main Page.

Fig. 4 to Fig. 6 illustrate the interfaces of SeRIS. The system was implemented to validate the propose ageing measurements and automate the ageing process to ensure the correctness of the computational that involve in the model. The SeRIS prototype system assists users in applying SEANA model in the real environment. SeRIS provides interface to input data of the targeted application software to be evaluated, computes the ageing scores for each factors and produces the ageing level and report.

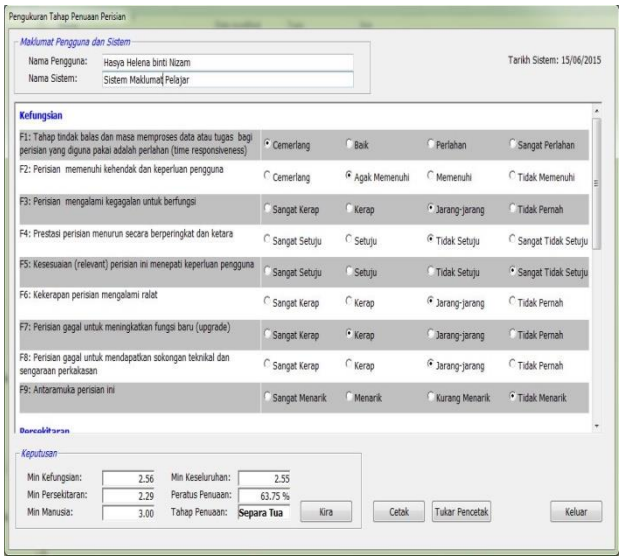

Fig 5. SeRIS Main Page - After Assessment.

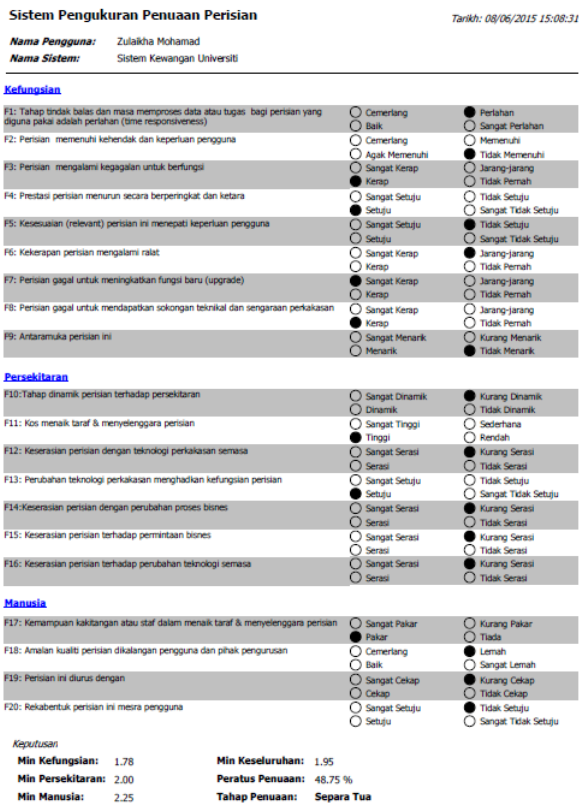

Fig 6. Assessment Report Generated by SeRIS System.

VI. The Anti-Ageing Guidelines and Actions

A. The Anti-Ageing Guideline

The next step in the anti-ageing process is to identify the necessary anti-ageing actions associated with each of the ageing factors. Based on the ageing index level shown in Table III, the anti-ageing actions are proposed to be carried out and applied on the software that has the lowest score during the ageing assessment described in Section IV. The anti-ageing actions are derived from software maintenance activities. Previous researchers revealed that one way to manage software ageing was through systematic maintenance of the software [20]. Software maintenance is a vast activity process of modifying and upgrading the software or part of the software to repair software error or fault, to add new functions or modification, or adaptation to a new environment [22][23].

As discussed in Section II, software maintenance activity can be categorized into four different types: adaptive, corrective, perfective and preventive. This research adopts the maintenance approaches as defined by Matias et al [14] as the baseline of the anti-ageing actions. The proposed anti-ageing actions are derived from literature and case study findings. The actions are recommended to ensure the software stays relevant and fulfil users' need and expectation in the dynamic environment today for longer time of usage. Table VII-IX show the anti-ageing guideline and actions for functional, environment and human factor related to the ageing factors and measurements.

TABLE VII. Anti-Ageing for Functionality Factors

	Metrics	Anti-Ageing Action
1	Time responsiveness	*Perfective* - Monitor the memory usage - Improve the quality aspect of the software. - Check the software structure (optimisation) *Adaptive* - Check quality assessment based on user's perspective and approach. - Enhance or modify software based on user requirements and expectations regularly.
2	Users' requirement and expectation	
3	Functionality	*Corrective* - Correct the error and fault of the software accordingly and systematically.
4	Degradation of Performance	*Perfective* - Improve the functional of software service to increase performance *Corrective* - Check and correct faults tolerant and failure regularly
5	Software Relevancy	*Perfective* - Improve and enhance the software function to ensure the up-to-date service/functions are available.
6	Software Faults and Failures	*Corrective* - Correct the fault and error accordingly and systematically. - Improve the change request and process.

TABLE VIII. Anti-Ageing for Environmental Factors

	Metrics	Anti-Ageing Action
1	Dynamic to environment change	*Corrective* - Easy maintenance for environment change - Easy maintenance for business change.
2	Cost for software maintenance and software upgrading	*Training* - Focus on in-house training for staff - Minimize outsourcing - In-house maintenance
3	Technology demand and compatibility	*Adaptive & Perfective* - Improve and enhance the software according to new/current technology demand and compatibility.
4	Hardware Changes	*Adaptive* - Improve and enhance the software for meeting new/current hardware demand and compatibility.
5	Business process change and demand	*Adaptive* - Improve and enhance the software function/services to ensure the demands in business processes are maintained and achieved.
6	Software technology change	*Adaptive* - Enhance the software for meeting new/current software technology demand and compatibility.

TABLE IX. Anti-Ageing for Human Factors

	Metrics	Anti-Ageing Action
1	Upgrading and maintenance expert	*Training* - Focus on training for staff in software maintenance and related.
2	Software quality practice	*Training* - Awareness to the staff for quality assurance - Train staff for software quality practices and implementation.
3	Software management capability	*Training* - Train for software management practices.
4	User Interface	*Adaptive* - Improve and enhance the software for user friendliness and usability aspect - Improve and enhance software functionality.

B. The Anti-Ageing Action Implementation

The proposed anti-ageing actions are suggested to the systems that have been tested in the case study discussed in previous section. Based on the assessment results, we choose the lowest score among the three systems. In this case the App1 system has been selected with the lowest percentage (47%) and ageing index level of Semi-old. The implementation of the anti-aging actions has been carried out on the three metrics that obtained the lowest score during the assessment. Table X shows the metrics and the proposed anti-ageing actions.

TABLE X. IMPLEMENTATION OF ANTI-AGEING FOR ACTION

Metrics	Score (1..4)	Anti-Ageing Action
Time responsiveness	1	Refer to Table VII to apply the anti-ageing action for time responsiveness. Suggested action to be applied is perfective maintenance.
User Interface	1	Refer to Table IX to apply the anti-ageing action for User Interface. Suggested action to be applied is adaptive maintenance.
Cost for software maintenance and software Upgrading	1	Refer Table VIII to apply the anti-ageing action for reducing the cost for software maintenance and software upgrading. Suggested action to be applied is by conducting in-house training for manintenance and try to minimize outsourcing.

VII. DISCUSSION

This research focuses on identification of software ageing factors and measurements as the fundamental of further related research which in our scope is the anti-ageing model, guideline and actions. This paper starts the discussion with presenting the background and related work of software ageing. The underlying issues and works related to software ageing, rejuvenation, anti-ageing and green and sustainability have been investigated and studied. Later we conducted the empirical study to explore more from real industrial perspective on these related issues and topics.

The findings from empirical study were used as the input to the development of the anti-ageing model. This model is called **S**oftwa**re An**ti-**A**geing or SEANA model. The main objective of the survey was to validate and verify the ageing factors among software practitioners. The empirical study which was conducted in Malaysia revealed three main ageing factors and associated measurements. The ageing factors are categorized as functional, environment and human. Based on the analysis, twenty metrics have been recognized to measure software ageing. The metrics were assigned with numerical scales for further quantifying of the software ageing score and level.

The SEANA model was developed as shown in Fig. 3 and could be used to measure the ageing status/level of any application software operating in certain environments. After the ageing level has been identified based on the assessment, the anti-ageing actions can be generated further aligned with the results of the ageing index. The anti-ageing actions are proposed in order to counteract and minimize the occurrence of ageing phenomenon in the specific targeted software. This is believed will prolong the relevancy of the software operating in their environments. The anti-ageing guideline and actions defined in this model will assist and ease the software owner or the stakeholder to make decision on the solution to be taken in order to deal with software ageing phenomenon if it occurs in their applications.

The SEANA model has been validated and applied in real case study collaborated with local industry in Malaysia. In this agency, three application software were used as case study as described in this paper. Furthermore, the prototype system, SeRIS was developed to validate and automate the process. The case study and SeRIS prototype system prove the effectiveness and practicality of SEANA model.

VIII. CONCLUSIONS

As a conclusion, even though software ageing is inevitable it can be delayed by applying anti-ageing techniques that has been presented in this paper. This study focuses on external factors of software ageing and identifies three main and essential ageing factors which are environment, human and functional. The anti-ageing model for application software was developed and tested in real industrial environment, and further recommended an anti-ageing guideline and actions associated with ageing phenomenon in software. For future work, it is suggested that the anti-ageing model proposed in this research to be applied and aligned with the green and sustainability context of software product. Sustainability dimensions which are social, economy and environment can be embedded in the new enhance anti-ageing model. Furthermore, with the prolong usage of software and delaying the aged of the software will reduce the waste and maintain minimum waste disposal of software product and development process.

ACKNOWLEDGMENT

This work was supported in part by the Arus Perdana Grant of UKM (AP-2017-005/3).

REFERENCES

[1] Li, Y. Qi, and L. Cai, “A Hybrid Approach for Predicting Aging-Related Failures of Software Systems,” 2018 IEEE Symp. Serv. Syst. Eng., pp. 96–105, 2018.

[2] J. H. Yahaya and A. Deraman, “Towards the Anti-Ageing Model for Application Software,” Proc. World Congr. Eng., vol. II, 2012.

[3] L. Parnas, “Software Aging Invited,” ICSE ’94 Proc. 16th Int. Conf. Softw. Eng., pp. 279–287, 1994.

[4] S. Ahamad, “Study of Software Aging Issues and Prevention Solutions,” Int. J. Comput. Sci. Inf. Secur., vol. 14, no. 08, pp. 307–313, 2016.

[5] Melo, J. Araujo, V. Alves, and P. Maciel, “Investigation of software aging effects on the OpenStack cloud computing platform,” J. Softw., vol. 12, no. 2, pp. 125–138, 2017.

[6] Cotroneo, A. K. Iannillo, R. Natella, R. Pietrantuono, and S. Russo, “The software aging and rejuvenation repository: Http://openscience.us/repo/software-Aging/,” 2015 IEEE Int. Symp. Softw. Reliab. Eng. Work. ISSREW 2015, pp. 108–113, 2016.

[7] R. Mohan and G. Ram Mohana Reddy, “Software aging trend analysis of server virtualized system,” Int. Conf. Inf. Netw., pp. 260–263, 2014.

[8] J. H. Yahaya, Z. N. Zainal Abidin, and A. Deraman, “Software Ageing Measurement and Classification Using Goal Question Metric (GQM) Approach,” Sci. Inf. Conf. 2013, pp. 160–165, 2013.

[9] Y. Zhao, J. Xiang, S. Xiong, Y. Wu, J. An, S. Wang, and X. Yu, “An Experimental Study on Software Aging in Android Operating System,” 2015 2nd Int. Symp. Dependable Comput. Internet Things, pp. 148–150, 2015.

[10] J. Araujo, V. Alves, D. Oliveira, P. Dias, B. Silva, and P. Maciel, “An Investigative Approach to Software Aging in Android Applications,” 2013 IEEE Int. Conf. Syst. Man, Cybern., pp. 1229–1234, Oct. 2013.

[11] S. Huo, D. Zhao, X. Liu, J. Xiang, Y. Zhong, and H. Yu, “Using Machine Learning for Software Aging Detection in Android System,” 2018 Tenth Int. Conf. Adv. Comput. Intell., pp. 741–746, 2018.

[12] J. Araujo, R. Matos, V. Alves, P. Maciel, F. V. de Souza, R. M. Jr., and K. S. Trivedi, “Software aging in the eucalyptus cloud computing infrastructure,” ACM J. Emerg. Technol. Comput. Syst., vol. 10, no. 1, pp. 1–22, 2014.

[13] J. H. Yahaya, A. Deraman, S. R. A. Ibrahim, and Y. Y. Jusoh, "Software Certification Modeling: From Technical to User Centric Approach" Aust. J. Basic Appl. Sci., vol. 7, no. 8, pp. 9–18, 2013.

[14] R. Matias Jr., K. S. Trivedi, and P. R. M. Maciel, "Using Accelerated Life Tests to Estimate Time to Software Aging Failure," 2010 IEEE 21st Int. Symp. Softw. Reliab. Eng., pp. 211–219, Nov. 2010.

[15] Z. H. Abdullah, J. Yahaya & A. Deraman "The Anti-Ageing Model for Assessment of Application Software," Postgrad. Res. Work. , SoftTech Asia 2018, 2018.

[16] F. Machida, J. Xiang, K. Tadano, and Y. Maeno, "Lifetime extension of software execution subject to aging," IEEE Trans. Reliab., vol. 66, no. 1, pp. 123–134, 2017.

[17] D. Cotroneo, R. Natella, R. Pietrantuono, "A Survey of Software Aging and Rejuvenation Studies," ACM J. Emerg. Technol. Comput. Syst. - Spec. Issue Reliab. Device Degrad. Emerg. Technol. Spec. Issue WoSAR 2011, vol. V, no. 212, p. 30, 2010.

[18] D. N. Arnold, "The Patriot Missile Failure," 2000.

[19] D. Cotroneo, R. Natella, R. Pietrantuono, and S. Russo, "Software Aging and Rejuvenation: Where We Are and Where We Are Going," 2011 IEEE Third Int. Work. Softw. Aging Rejuvenation, no. 30, pp. 1–6, Nov. 2011.

[20] J. Zhao, K. S. Trivedi, Y. Wang, and X. Chen, "Evaluation of software performance affected by aging," 2010 IEEE Second Int. Work. Softw. Aging Rejuvenation, vol. 3, pp. 1–6, Nov. 2010.

[21] I. Sommerville, Software Engineering (Tenth Edition). 2016.

[22] Z. H. Abdullah, J.H.Yahaya, Z. Mansor & A.Deraman. "Software Ageing Prevention from Software Maintenance Perspective – A Review," Journal of Telecommunication, Electronic and Computer Engineering, vol. 9, no. 3-4, pp. 93-96, 2017.

[23] J. H. Yahaya, A. Deraman & Z. H. Abdullah. "Evergreen Software Preservation: The Conceptual Framework of Anti-Ageing Model," Information Science and Applications, Lecture Notes in Electrical Engineering, vol. 339, pp 899-906, 2015.

[24] Z. N. Zainal Abidin, J.H. Yahaya, A. Deraman & Z. H. Abdullah. "Rejuvenation Action Model for Application Software," The 6th International Conference on Information and Communication Technology (ICoICT 2018), 3-5 May 2018.

[25] S. Murugesan & P.A. Laplante. "IT for a greener planet," IT Pro January/February, pp. 16–20, 2011.

[26] Murugesan, S. "Harnessing green IT: Principles and practices," IEEE IT Professional, vol. 10, no. 1, pp.5-6, 2008.

[27] M. Dik, J. Drangmeister, E. Kern & S. Naumann. "Green software engineering with agile methods in green and sustainable software (GREENS)," Proceedings of 2013 2nd International Workshop on Green and Sustainable Software , pp 78–85, 2013.

[28] K. Erdelyi. "Special factors of development of green software supporting eco sus-tainability," Proceedings of EEE 11th international symposium on intelligent systems and informat-ics (SISY), pp 337–340, 2013.