The Journal of Supercomputing (2021) 77:8151–8171
https://doi.org/10.1007/s11227-020-03610-6

A method of activity-based software maintenance cost estimation for package software

Kyoung-ae Jang[1] · **Woo-Je Kim**[1]

Accepted: 28 December 2020 / Published online: 14 January 2021

Abstract
This paper defines software maintenance activities and develops a model for maintenance cost estimation of package software. First, we classified software maintenance activities which were collected from the literature reviews. Second, we developed a cost structure for package software maintenance based on the identified activities. Third, the activity-based software maintenance cost estimation model was developed based on the software maintenance activities and cost structure. Fourth, we defined the adjustment factors for the software maintenance cost estimation model to improve the accuracy of the developed model. Finally, the developed model was verified using actual data from software maintenance projects. The proposed model showed reliable performance in terms of the mean magnitude of relative error and prediction quality. Therefore, the proposed model is suitable for setting budgets and predicting costs associated with package software maintenance projects.

Keywords Software maintenance · Cost estimation · Activity · Cost model · Package software

1 Introduction

The proportion of software in systems is increasing, and software has become an indispensable factor. A 2010 survey showed that approximately 17.6% of software sales in package software companies in Korea were related to the maintenance of package software [1]. Revenue from package software consists of license costs and maintenance costs. The cost of licensing the product is dependent on the maintenance cost policy. The software maintenance cost is an important factor in the package software revenue model.

✉ Woo-Je Kim
wjkim@seoultech.ac.kr

[1] Department of Industrial Engineering, Seoul National University of Science and Technology, 232, Gongneung-ro, Nowon-gu, Seoul 01811, Republic of Korea

Although the package software sector is growing, there are insufficient standards in the software maintenance market of the sector. It is difficult to standardize software maintenance activities because there exist several business activities that are related, and there is no clear demarcation between defect repair, maintenance, and free servicing activities. Thus, clear software maintenance cost criteria need to be defined. To achieve this, we defined software maintenance activity elements for package software and developed an activity-based software maintenance cost estimation model.

Cost estimation is a valuable task for managers that helps them plan a software maintenance project and perform cost benefit analysis [2]. A considerable number of previous studies have been carried out to design cost estimation models of software maintenance [2–6], but most of them have focused on specific customer projects such as in-house systems, not package software. Therefore, there is a need for research on software maintenance cost estimation for package software. In the usual scenario, a project owner estimates the software maintenance cost by proportionally considering the specified rate to the purchasing price of the package software in Korea when he/she sets a budget for a software maintenance project for package software. Usually the rate to the purchasing price for software maintenance cost estimation is quite low in Korea. Therefore, the companies that make package software have difficulty achieving profitability in Korea, which shows that there is a need to have a model to estimate the software maintenance cost of package software [7].

This paper defines software maintenance activities and proposes a model for software maintenance cost estimation of package software. First, we review the literature surveys to provide software maintenance activity classifications or activity factors. Then, we develop a cost structure for package software based on the identified activity factors, and propose an activity-based cost estimation model. Also, we develop adjustment factors for software maintenance cost, and completed the activity-based software maintenance cost estimation for package software. Finally, we verify the model with actual cost data from software maintenance projects.

The rest of this paper is structured as follows: Section 2 introduces the theoretical background for this study. Section 3 discusses the design of the model and verifies the developed model. Section 4 provides the conclusions and discusses future research.

2 Research background

2.1 Previous research

Planning a software maintenance project involves estimating size, effort, duration, staff, and costs, and it is a key factor for a successful software maintenance project [2]. To achieve an accurate estimate of the overall software maintenance cost, it is necessary to define each activity element, estimate individual costs, and combine these using a suitable model. Previous studies have extensively reviewed software maintenance activities such as error correction, capability enhancement, obsolete capability removal, and optimization [8–10]. It is important to note that software

maintenance cost depends on the size and types of work required, workforce capability and familiarity with the system, processes and standards in use, and complexity [3, 11, 12].

The relevant standards set by government agencies and standards associations are an important component of software maintenance projects. ISO/IEC 14764 defined software maintenance as the modification of a software product after delivery to correct faults and improve performance or other attributes [9]. ISO/IEC 14764 classifies four types of software maintenance activities based on ISO/IEC 12,207 components [9], and the IEEE 1219 model is an extension of ISO/IEC 12207 [13].

The International Function Point User Group (IFPUG) studied software maintenance activities and proposed a cost estimation model based on function point (FP) [14]. Other models have been proposed by the International Software Benchmarking Standards Group (ISBSG) [15], United Kingdom Software Maintenance Association (UKSMA) [16], and Korea Software Industry Association (KOSA) [17]. The National IT Industry Promotion Agency (NIPA) has also proposed definitions of maintenance activities for open source software in Korea [18], and the Korea Internet and Security Agency (KISA) proposed a model to estimate the maintenance cost of information security systems [19].

Mather [20] studied software maintenance costs, and Stark [21] studied software maintenance factors. Hunt [22] investigated cost impact factors through case studies. Package software maintenance service guide refers to activities such as product services, technical services, and user support services to utilize and maintain the software in an optimal state. The guide proposed two types of software maintenance cost estimation systems: flat fee system and variable rate system [23]. The variable rate system incorporates license cost and maintenance servicing cost explicitly, whereas the flat fee system has an annual cost, including license cost and maintenance service cost. This guide suggests that the contractor and contractee adjust the cost fee by mutual consultation. A guide for cost estimation of a software project suggests a package software maintenance rate of 15% to the licensing price, but it does not provide a cost estimation method related to the maintenance activities [17].

2.2 Delphi and Kawakita Jiro method

This study applied the Delphi technique to gather opinions from anonymous online experts and the Kawakita Jiro (KJ) method to classify similar items subsequently. The Delphi technique is commonly used to solve uncertain and difficult problems through an integrated opinion of a group, rather than individual opinions [24]. The Delphi technique provides impartial and objective feedback by ensuring impartiality through the anonymous participation of experts. The KJ method involves expert group off-line discussion [25]. The KJ method involves the problem being written on a paper card with similar response items integrated and named. The analysis phase of the KJ method consists of the following five steps: target list definition, recording (on the paper card), expert opinion exchange and grouping of responses, name definition of the grouped responses, and developing the affinity diagram.

This paper defines software maintenance activities of package software using the Delphi technique and literature surveys, and integrates the activity elements using the KJ method.

3 Research design and research results

3.1 Classification of software maintenance activities

ISO/IEC 14,764 categorizes software maintenance activities into four categories: maintenance, requests, corrections, and enhancement [9]. IEEE 1219 defined the categories of software maintenance activities with corrective, adaptive, perfective, and emergency maintenance, where the corrective maintenance and perfective maintenance were the same as activities in ISO/IEC 14,764, and the emergency maintenance meant correcting unforeseen errors for continued operation [13]. IFPUG classified software maintenance activities with corrective, adaptive, and preventive/perfective maintenance [14]. ISBSG classified them as enhancement, maintenance, support, and operation [15] and UKSMA defined two categories: perfective/preventive and corrective maintenance [16].

In Korea, KOSA proposed software maintenance activities and a maintenance cost estimation model for in-house software [17], and NIPA categorized the activities for security software as preventive, adaptation, repair, support, and daily maintenance [18]. KISA classified them as pattern, certification, improvement, and technical support services [19]. Open source software maintenance activities were classified as installation services, patch delivery, upgrade, optimization, tuning, troubleshooting, monitoring, online support, technical consulting, and warranty in a guide for maintenance service of open source software [18]. In addition, product services, technical services, and user support services were elicited as software maintenance services in the package software maintenance service guide [23].

This paper classified the software maintenance activity elements by integrating complementary aspects of these various studies. Table 1 shows a classification of the software maintenance activities derived from the previous studies by literature surveys.

Among the previous studies, the package software maintenance service guide was well-organized. Therefore, we integrated the activities based on the guide and classified the activities into product service, technology service, and support service using the KJ technique. Some of the criteria we integrated into the activity areas follows:

- Pattern maintenance includes error correction and version patch of the package software.
- Certification includes services to quantify how well the package software meets the quality assurance plan.
- Improvement services include adaptive patch service and corrective patch service, and developing new defenses for hacking.

Table 1 Software maintenance service activities for package software

Package software [22]	Open software [17]	Security software [18]	ISO/IEC 14,764 [8]	IEEE 1219 [12]	IFPUG [13]	KOSA [16]
Patch service	Patch service	Pattern maintenance	Adaptive	Adaptive	Adaptive	Adaptive
Update	Update	Pattern Maintenance	Error correction	Error correction	Error correction	Correction
Upgrade	Upgrade	Modification development	–	–	–	–
Daily support	Monitoring	Technical service	Perfective	Perfective	Preventive/perfective	Daily/perfective
Daily support	Help online	Technical service	Perfective	Perfective	Preventive/perfective	Daily/perfective
Daily support	Developer support	Technical service	Perfective	Perfective	Preventive/perfective	Daily/perfective
Daily support	Technical advisory	Technical service	Perfective	Perfective	Preventive/perfective	Daily/perfective
(Urgent) error handling service	Installation (reinstallation)	Technical service	–	Urgent service	Preventive/perfective	–
(Urgent) error handling service	Optimization	Technical service	–	Urgent service	Preventive/perfective	–
(Urgent) error handling service	Tuning	Technical service	–	Urgent service	Preventive/perfective	–
(Urgent) error handling service	Problem Solving	Technical service	–	Urgent service	Preventive/perfective	–
Preventive/predictive support	Update	Technical service	Preventive	–	Preventive/perfective	–
Preventive/predictive support	Optimization	Technical service	Preventive	–	Preventive/perfective	–
Preventive/predictive Support	Tuning	Technical service	Preventive	–	Preventive/perfective	–
Operator/user training	Developer Support	Technical service	–	–	–	–
Quality assurance and certification	Assurance	Assurance	–	–	–	–

Table 2 Definition of software maintenance service activities for package software

First item	Second item		Definition
Product	Product modification and complement	Adaptive patch	Development and distribution of patches to accommodate changes in operating environment such as operating system changes
		Corrective patch	Development and distribution of patches to fix bugs and other errors in existing software products
Technology	Online support		Remote daily support such as daily inquiries via phone/email, online, and remote monitoring service
	Emergency visit service		Serviceman's visiting service to fix breakdown of the package software. It includes software reinstallation service, maintenance service to solve urgent problem, and so on
	Regular visit service		Serviceman's regular visiting service to prevent system failure in advance. It includes periodic inspection, regular performance tuning service, and service to visit for regular maintenance
Support	Training		Operator/user training for operation and use of product
	Quality assurance and certification		Support for legal issues related to the use of software products, activities for quality assurance and certification

- Technical services include disaster recovery, help desk support, and preventative maintenance.

We developed a standard model for software maintenance activities based on the above criteria and the previous studies. The model has product service, technology service, and support service. The product service consists of an adaptive patch service and a corrective patch service. The technology service consists of online support, emergency visit service, and regular visit service. The support service consists of training for operation and quality assurance and certification. Table 2 shows the refined classification for software maintenance service activities of package software.

3.2 Software maintenance cost estimation model

3.2.1 Model development

The software maintenance costs are made up of the direct labor cost, indirect overhead expense, engineering fee, direct overhead cost, and value added tax (VAT).

The direct labor cost is the major component of software maintenance cost, and it is the direct cost of the workforce to perform the software maintenance activities.

The indirect overhead expense refers to the overhead cost not directly associated with software maintenance activities, such as planning costs, administrative expenses, and costs for general affairs. According to the engineering cost estimation model in Korea, the overhead expense can be estimated as110–120% of the direct labor cost [16, 26].

The engineering fee means the costs of research and investigation, technology development, technical education, and margins. According to the engineering cost model in Korea, the engineering fee can be estimated as 20–40% of the sum of the direct labor cost and the indirect overhead expense [16, 26].

The direct overhead cost and VAT should also be considered. The direct overhead cost is the direct cost required to perform the software maintenance project such as the traveling expense, office rental fee, and printing fee. The direct overhead cost is empirically estimated by the project team. The VAT is assumed to be 10% of the total amount in Korea. Figure 1 shows the proposed software maintenance cost structure model for package software.

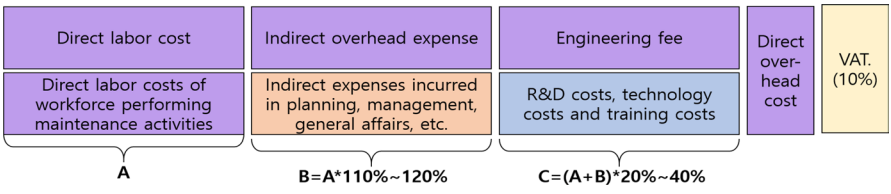

Fig.1 Software maintenance service cost structure model

Therefore, the total cost for a package software maintenance project is obtained by summing the direct labor cost, the indirect overhead expense, the engineering fee, the direct overhead cost, and the VAT. Therefore, the equation for the total software maintenance cost is

$$\begin{aligned}\text{Total cost} &= \text{Direct labor cost} + \text{Indirect overhead expense} + \text{Engineering fee} \\ &\quad + \text{Direct overhead cost} + \text{VAT}\end{aligned} \tag{1}$$

As the software maintenance activities mainly include product services such as adaptive patch services and corrective patch services, technology services which include online technical support services, emergency visit services, and regular visit services, and support services which are training services and quality assurance and certification services as shown in Table 2, the direct labor cost can be obtained by summing the cost components required to perform the adaptive patch services, corrective patch services, online support services, emergency visits, regular visits, and support services The direct labor cost components can be individually estimated based on each software maintenance service activity.

Each direct labor cost component can be derived by the number of occurrences for each service activity, the average service time for each service activity, the number of workers required to perform the service activity, and the manpower level for each service activity. The number of occurrences, the average service time, the number of workforces required, and the manpower level for each service activity can be obtained from historical records, which have been accumulated from previous software maintenance projects. Normally, as a software maintenance project is carried out every year, historical records for each software maintenance activity are accumulated year after year. Therefore, those factors can be obtained from the historical records and modified to the current software maintenance project. However, if a software maintenance project is being carried out for the first time, the required values should be empirically estimated by experts. Depending on the information on the manpower level for each software maintenance service activity, the unit cost of wage can be obtained, as there is a standard unit cost for the wage for each level of manpower. Hence, the direct labor cost for each software maintenance service activity can be obtained by multiplying the number of occurrences, the average service time, the number of people required, and the unit cost of wage. Furthermore, the total direct labor cost is the sum of the individual direct labor costs for each software maintenance service activity.

The total direct labor cost can be obtained by the following equation:

$$\begin{aligned}\text{Direct labor cost} &= \text{the number of occurrences} \times \text{the average service time} \\ &\quad \times \text{the number of workforce} \times \text{unit cost of wage for level of workforce}\end{aligned} \tag{2}$$

As shown in Eq. (1), the total cost for a software maintenance project is composed of the direct labor cost, the indirect overhead expense, the engineering fee, the direct overhead cost, and VAT. According to the engineering cost estimation model in Korea, the indirect overhead expense is 110–120% of the direct labor cost, and the engineering fee is 20–40% of the sum of the direct labor cost and the indirect

overhead expense [26]. If we apply 110% of the direct labor cost for the indirect overhead expense and 20% of the sum of the direct labor cost and the indirect overhead expense for the engineering fee, the equation for the total cost of the software maintenance project can be summarized as

$$\begin{aligned}\text{Total cost} &= \text{Direct labor cost} + 1.1 \times \text{Direct labor cost} + 0.2 \times (\text{Direct labor cost} \\ &\quad + \text{Indirect overhead expense}) + \text{Direct overhead cost} + \text{VAT} \\ &= \text{Direct labor cost} + 1.1 \times \text{Direct labor cost} + 0.2 \times (\text{Direct labor cost} \\ &\quad + 1.1 \times \text{Direct labor cost}) + \text{Direct overhead cost} + \text{VAT}\end{aligned} \tag{3}$$

$$\text{Total cost} = 2.52 \times \text{Direct labor cost} + \text{Direct overhead cost} + \text{VAT} \tag{4}$$

Thus, we can estimate the software maintenance cost of package software by only estimating the direct labor cost and the direct overhead cost.

The direct labor cost is then estimated for each software maintenance service activity based on the number of occurrences, the average service time for the software maintenance activity, the number of workers required for the software maintenance activity, and the unit cost of wage related to the level of workers to carry out the software maintenance activity. The values for the cost components are obtained from previous software maintenance project experience, and can be modified to match the individual situation.

For example, we estimate the direct labor cost for the corrective patch service. First, we assumed that the corrective patch services occurred 12 times per year, and the average service time for one corrective patch was 1 h, the number of workers required for the corrective patch service was 2, and the level of the worker needed to carry out the corrective patch service was highly qualified persons. Furthermore, the unit cost of wage for the highly qualified worker was assumed to be 336 dollars/(person, day) × 1/8 h/day = 42 dollars/(person, hour). Then, the direct labor cost for the error patch service can be calculated by 12.0 times/year × 1.0 h/case × 2.0 persons/case × 42 dollars/(person, hour) = $1,008.

Table 3 Template for the direct labor cost based on software maintenance activities

Name of software maintenance project				Project A		
Activity	Corrective patch	Adaptive patch	Online support	Urgent visit	Regular visit	Support
Number of occurrences	5.0	6.0	3.0	10.0	12.0	4.0
Average service time	3.0	2.0	0.5	2.0	1.0	1.0
Number of workforce	2.0	2.0	1.0	2.0	1.0	1.0
Workforce level	High	Middle	High	High	Middle	Middle

The total direct labor cost is estimated by summing the costs for the adaptive patch service, corrective patch service, online support service, emergency visit service, regular visit service, and support service.

The template for obtaining information on cost components that are included in the direct labor costs by the software maintenance activity is developed as shown in Table 3. The number of occurrences, the average service time, the number of workers, and the level of the workforce are collected for the software maintenance project using Table 3.

If we assume that the unit cost of wage for high class workers is 42 dollars/(person, hour) and that for middle class workers is 32 dollars/(person, hour), the direct labor cost for the corrective patch service is \$1,260 ($5\times3\times2\times42$), that for the adaptive patch service is \$768 ($6\times2\times2\times32$), that for the online support service is \$63 ($3\times0.5\times1\times42$), that for the urgent visit service is \$1,680 ($10\times2\times2\times42$), that for the regular visit service is \$384 ($12\times1\times1\times32$), and that for the support service is \$128 ($4\times1\times1\times32$).The total direct labor cost for the above software maintenance project is \$4,283 (\$1,260 + \$768 + \$63 + \$1,680 + \$384 + \$128).

If we assume that the direct overhead cost is \$1,000, the indirect overhead expense is 110% of the direct labor cost, and the engineering fee is 20% of the sum of the direct labor cost and the indirect overhead expense, the total software maintenance cost, including the VAT, for the above project is \$12,972.48 [$(2.52\times\$4{,}283+\$1{,}000)\times1.1$].

3.2.2 Development of adjustment factors for software maintenance cost

We have furthermore studied the method of applying the adjustment factors to lower the estimation error rate and raise the accuracy in the developed software maintenance cost estimation model. Software maintenance cost is affected by the characteristics of the software, several environmental factors during the software production and maintenance process. The software maintenance cost can be affected by the project size, the capacity of the project workforce, the capabilities of the enterprise, the complexity of the software [27–32]. In a previous study, Mehdi Hejazi Dehaghani et al. [27] selected 32 components to determine the software maintenance cost and defined the weights of the components using analytic hierarchy process (AHP) method. They selected attributes of factors of project maintenance that are software experience, application understanding, software development reference access to techniques, document quality, software aging, group capability, environment dependency, and structure independence. Rajiv D. Banker et al. [28] proposed function point complexity, project management factors, user factors, and technical environment factors as factors affecting software maintenance productivity using data envelopment analysis (DEA) technique. Edward E. Ogheneovo [29] studied the relationship between software complexity and software maintenance cost, and found that software maintenance cost would be increased as software became more complex and bugs increased. Nexhati Alija [30] proposed cost factors of software maintenance such as team stability, contractual responsibility, staff skills, program age, program structure,

Table 4 Definition of adjustment factors of software maintenance cost

First group	Adjustment factor	Previous studies
Project management [28]	Software age	Software old [27], program age and structure [30]
	Level of documentation	Document quality [27], document level [32]
Technical environment [28]	Developer experience	Staff skills [30], software experience [27], application understanding [27]
	Enterprise maturity level	Group capability [27], team stability [30], contractual responsibility [30]
	Level of software complexity	Software complexity [29], complexity [31], function point complexity [28], software complexity [32]

Table 5 Define the weights of adjustment factors

Adjustment factors	Criterion for weight	Weight	Adjustment factors	Criterion for weight	Weight
Software age (SA)	< 1 year	1.1	Enterprise maturity level (EM)	> 10 years	1.1
	1 year ~ 3 years	0.9		5 years ~ 10 years	1
	> 3 years	0.7		< 5 years	0.8
Level of documentation (LD)	Excellent	1.2	Level of software complexity (SC)	For business assistant	0.9
	Good	1		For business processing	1
	Poor	0.9		For science and technology	1.1
Developer experience (DE)	> 5 years	1.1		For intelligence information	1.2
	3 years ~ 5 years	1		For system	1.3
	< 3 years	0.9		For communication control	1.4
				For process control	1.5
				For command and control	1.6

stressful sort of work, and fluctuations in the workload and resourcing problems in a study about the justification of software maintenance costs. Also, F. Niessink and H. Van Vlient [31] claimed that software maintenance costs should be estimated based on effort, and the effort could be affected by software complexity, the number of change requirements, and program size. George E. Stark [32] categorized software maintenance cost factors such as duration, software complexity, user satisfaction, and level of documentation.

As a result of analyzing the previous studies, the adjustment factors of software maintenance cost are developed by the KJ method as shown in Table 4. We derived five adjustment factors, namely software age, level of documentation, developer experience, enterprise maturity level, and level of software complexity.

Table 5 shows the weights of the structured adjustment factors which are obtained by the AHP method and criterion for weight. The weight of software age can be obtained based on how many years have passed since the software was developed. The criterion for the weight of level of documentation is composed of "excellent," "good," and "poor." The weight of developer experience can be obtained based on the number of years worked as a developer, and the weight of enterprise maturity level can be obtained based on the number of years since the company was established, in which the software is being used. The weight of level of software complexity can be obtained by the types of software approximately because the level of application complexity is estimated by the types of application in the guide of the Korean software cost estimation model [26].

3.3 Verification of the developed model

3.3.1 Verification of model performance before applying adjustment factors

The 19 software maintenance projects were selected in Korea to verify the performance of the developed model. From the software maintenance projects, the 19 actual data sets for cost components of software maintenance activities were collected. The 19 data sets have four basic cost components that are used to calculate the direct labor cost for each software maintenance service activity: the number of occurrences for each software maintenance service activity, the average service time for doing the service, the number of workers required to perform the software maintenance service activity, and the level of the workforce required to do the service activity. The actual data sets are summarized in Table 6.

With the actual data sets, the software maintenance cost for each software maintenance project is estimated by the developed model. The estimated cost obtained by the developed model was compared with the actual contract price of the software maintenance project to evaluate the performance of the model. Three

Table 6 Actual data sets of cost components for each software maintenance activity

Project		Project No.sss																		
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
CP[a]	NO[g]	5	88	1	1	10	6	6	120	5	7	7	2	1	6	4.8	24	24	5	24
	ST[h]	10	0.3	5	5	1	1	1	1.5	3	2	2	2	3	2	2	2	5	2	2
	NW[i]	3	1	5	3	3	1	1	1	2	4	4	1	2	5	1	2	1	1	2
	LW[j]	H	L	H	H	H	H	M	M	H	H	H	H	H	M	M	H	M	M	H
AP[b]	NO	5	0	2	2	2	6	6	40	6	10	10	5	6	6	1.2	6	1.2	1	6
	ST	5	0	20	15	1	1	1	5	2	1	1	1	3	2	5	5	20	5	5
	NW	3	0	5	3	1	2	2	1	2	1	1	1	1	5	1	1	2	1	1
	LW	M	-	H	H	M	H	M	M	M	M	M	M	M	M	M	H	H	M	H
OS[c]	NO	24	107	5	4	50	36	36	240	3	30	50	120	20	30	0	0	0	0	0
	ST	1	1	1	1	0.5	1.5	1.5	2	0.5	0.5	0.5	0.5	0.5	0.5	0	0	0	0	0
	NW	2	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
	LW	M	H	M	M	H	M	M	M	H	H	H	M	M	L	–	–	–	–	–
UV[d]	NO	4	0	2	1	5	6	6	10	10	15	30	7	5	5	360	120	480	360	120
	ST	2	0	1	1	1	1	1	2	2	0.5	0.5	1	1	1	0.5	0.2	0.2	1	0
	NW	1	0	2	2	3	1	1	1	2	1	1	1	1	1	1	1	1	1	1
	LW	M	-	H	M	H	M	M	M	H	H	M	H	M	M	M	M	M	M	-
RV[e]	NO	4	16	4	4	12	12	12	12	12	12	12	12	12	12	0	0	0	0	0
	ST	1	1	1	1	1	1	1	1	1	0.5	0.5	1	0.5	0.5	0	0	0	0	0
	NW	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
	LW	L	M	M	L	M	M	M	M	M	M	M	M	M	L	-	-	-	-	-
SS[f]	NO	4	77	1	2	5	2	2	4	4	5	5	6	1	2	0	0	0	0	0
	ST	1	1	1	1	2	1	1	1	1	1	1	2	1	1	0	0	0	0	0
	NW	2	2	1	1	0.5	1	1	1	1	2	2	1	1	1	0	0	0	0	0
	LW	M	H	L	L	L	M	M	M	M	M	M	M	L	H	–	–	–	–	–

CP[a]: Corrective patch service, AP[b]: Adaptive patch service, OS[c]: Online support service, UV[d]: Urgent visit service, RV[e]: Regular visit service, SS[f]: Support service, NO[g]: Number of Occurrences(times/year), ST[h]: Average service time(hour/case), NW[i]: Required number of workforce(person/case), LW[j]: Level of required workforce (High, Middle, Low)

measurements were used to verify the model: the coefficient of determination (R^2), magnitude of relative error (MRE), and prediction quality (PRED) measure. R^2 represents the rate of change in the dependent variable with the change in the independent variable, and is defined on the range [0, 1]; the larger the R^2 value of the model, the better it is fit. The magnitude of the relative error for the *i*th observed value (MRE_i) is defined as

$$\mathrm{MRE}_i = \frac{\left| y^*_{i-} y_i \right|}{y_i} \tag{5}$$

where y_i is the actual cost for the *i*th software maintenance project in the actual cost data set, and y^*_i is the estimated value for the *i*th software maintenance project using the proposed model. The mean of the MRE of the *N* estimates is

$$\mathrm{MMRE}_i = \frac{1}{N} \sum_{i=1}^{n} \mathrm{MRE} \tag{6}$$

The smaller the MRE of the model, the better it is. The third measure is the PRED for a given threshold, *p*, which is defined as the percentage of estimates where MRE is not greater than the threshold. It provides an indication of the overall fit of a set of data points, based on the MRE values for each data point. For example, if PRED (0.20) = 40%, then it means that 40% of the fitted values fall within 20% of their corresponding actual values.

$$\mathrm{PRED}\ (p) \ = \ k/N \tag{7}$$

where *k* is the number of estimates with MRE values failing in between 0 and p.

For the collected 19 data sets, the R^2 value between the estimated costs and actual contract prices is obtained in Fig. 2. In Fig. 2, the *x*-axis means the actual contract price and the *y*-axis means the estimated cost by the developed model. The R^2 value between the estimated costs and actual contract prices was 0.939.

Table 7 shows the MRE values obtained by the estimated cost and the actual contract price for the selected 19 software maintenance projects. The maximum MRE value was 43.91% and the minimum value was 0.61%. Furthermore, the mean MRE value (MMRE) was 16.69%.

Table 8 shows the measured performance for 19 actual cost data sets in PRED. The PRED (0.25) was 63%, the PRED (0.30) was 78%, and PRED (0.45) was 100%.

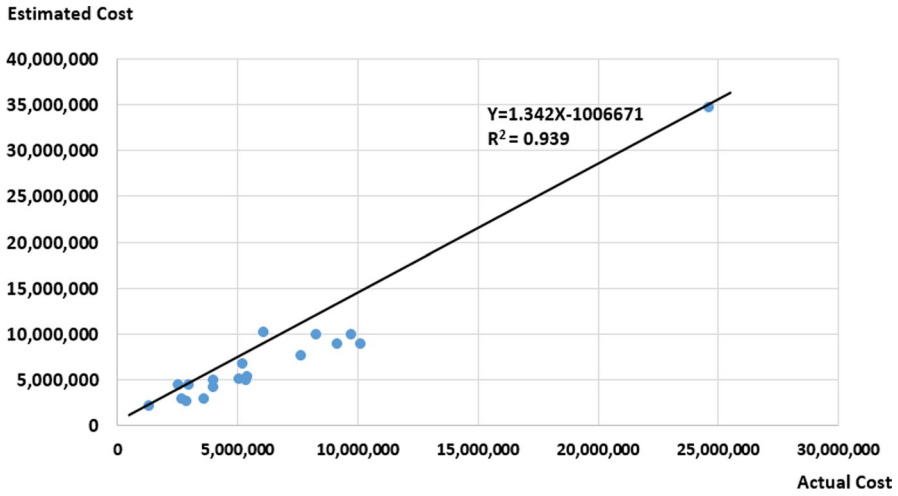

Fig.2 R^2 value before applying adjustment factors

Table 7 Magnitude of relative error for estimated cost and actual cost

No	Estimation cost	Actual cost	MRE (%)
1	9,129,000	9,000,000	1.43
2	10,104,000	9,000,000	12.27
3	8,253,000	10,000,000	17.47
4	3,954,000	5,000,000	20.92
5	2,959,000	4,500,000	34.24
6	2,658,000	3,000,000	11.40
7	2,524,000	4,500,000	43.91
8	24,583,000	34,682,000	29.12
9	3,590,000	3,000,000	19.67
10	5,193,000	6,800,000	23.63
11	6,035,000	10,200,000	40.83
12	2,825,000	2,772,000	2.59
13	1,301,000	2,200,000	40.86
14	3,948,000	4,250,000	7.11
15	5,367,000	5,400,000	0.94
16	5,052,000	5,105,000	1.04
17	7,600,000	7,702,500	1.30
18	9,688,000	9,950,000	7.64
19	5,335,000	5,000,000	6.70
MMRE			16.69

Table 8 PRED values obtained by the model for data set

Metrics	Value (%)	Metrics	Value (%)
PRED(0.20)	63	PRED(0.40)	84
PRED(0.25)	73	PRED(0.45)	100
PRED(0.30)	78	PRED(0.50)	100
PRED(0.35)	84	PRED(0.55)	100

Table 9 Results of MRE after applying adjustment factors

No	SA	LD	DE	EM	SC	F	G	H (%)	I (%)
1	1.0	1.0	1.0	1.0	0.9	9,129,000	9,042,582.0	1.43	0.47
2	0.9	1.0	1.0	0.9	1.0	10,104,000	9,006,962.1	12.27	0.08
3	1.0	1.0	1.0	1.0	1.1	8,253,000	9,990,949.1	17.47	0.09
4	1.0	1.0	1.0	0.9	1.3	3,954,000	5,091,278.8	20.92	1.83
5	1.0	1.0	1.0	1.0	1.4	2,959,000	4,517,109.4	34.24	0.38
6	1.0	1.0	1.0	1.0	1.0	2,658,000	2,925,221.6	11.40	2.49
7	1.1	1.0	1.0	1.0	1.5	2,524,000	4,583,765.5	43.91	1.86
8	1.1	1.0	1.0	1.0	1.2	24,583,000	35,713,512.2	29.12	2.97
9	0.9	1.0	1.0	0.9	0.9	3,590,000	2,880,645.3	19.67	3.98
10	1.1	1.0	1.0	1.0	1.6	5,193,000	6,683,366.5	23.63	1.72
11	1.1	1.0	1.0	1.0	1.6	6,035,000	7,649,383.7	40.83	25.01
12	0.9	1.0	1.0	1.0	1.0	2,825,000	2,798,597.6	1.91	2.83
13	1.1	1.0	1.0	0.9	1.5	1,301,000	2,125,653.8	40.86	3.38
14	1.0	1.0	1.0	1.0	1.0	3,948,000	4,345,435.2	7.11	2.25
15	0.9	1.0	1.0	1.0	1.0	5,367,000	5,315,718.6	0.61	1.56
16	1.0	1.0	1.0	0.9	1.0	5,052,000	5,003,717.2	1.04	0.07
17	1.0	1.0	1.0	0.9	1.0	7,600,000	7,527,826.0	1.33	0.37
18	1.0	1.0	1.0	1.0	0.9	9,688,000	10,191,178.3	2.63	2.42
19	1.0	1.0	1.0	1.0	1.0	5,335,000	4,834,979.8	6.70	3.30
MMRE								16.69	3

SA: Weight of software age, LD: Weight of level of documentation, DE: Weight of developer experience, EM: Weight of enterprise maturity level, SC: Weight of level of software complexity, F: Estimated cost before applying adjustment factors, G: Estimated cost after applying adjustment factors, H: MRE before applying adjustment factors, I: MRE after applying adjustment factors

3.3.2 Verification of model performance after applying adjustment factors

In order to increase the accuracy of the software maintenance cost estimation model, we conducted additional tests using the adjustment factors in the software maintenance cost estimation model. To do this, we investigated the weights of the adjustment factors for the 19 software maintenance projects as shown in Table 9.

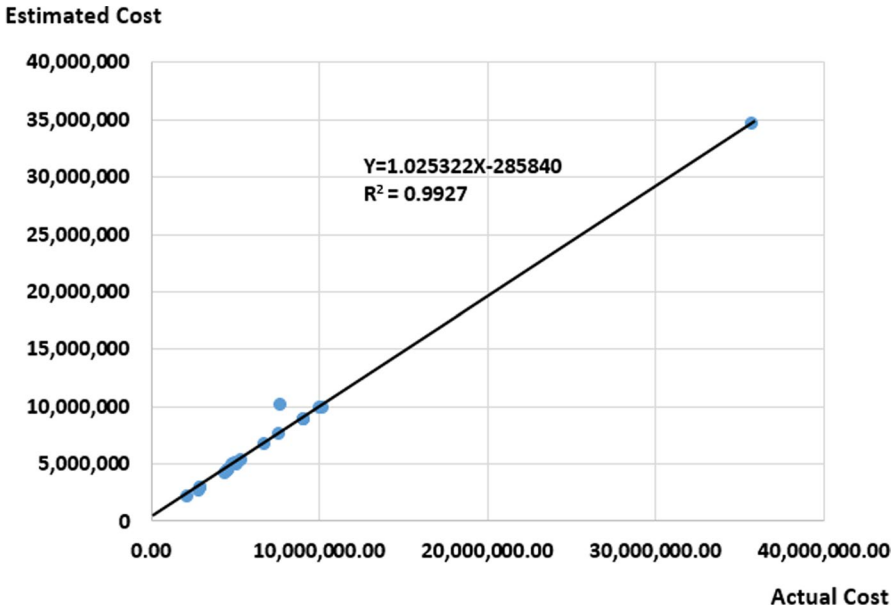

Fig.3 Results of R^2 after applying adjustment factors

Table 10 Results of PRED after applying adjustment factors

Metrics	Before applying adjustment factors (%)	After applying adjustment factors (%)
PRED(0.20)	63	94
PRED(0.25)	73	94
PRED(0.30)	78	100
PRED(0.35)	84	100
PRED(0.40)	84	100
PRED(0.45)	100	100
PRED(0.50)	100	100
PRED(0.55)	100	100

The multiple regression analysis was performed with the data of Table 9, and the adjusted software maintenance cost can be obtained by the following equation:

$$\begin{aligned}&\text{Adjusted software maintenance cost}\\&= \text{Estimation cost} \times \text{Weight of software age} \times \text{Weight of level of documentation}\\&\quad \times \text{Weight of developer experience} \times \text{Weight of enterprise maturity level}\\&\quad \times \text{Weight of level of software complexity} \times 0.0579(\text{Correction coefficient})\end{aligned} \tag{8}$$

As shown in Table 9, the maintenance cost of package software can be estimated by applying the adjustment factors. We compared the MREs of before applying adjustment factors and the MREs after applying adjustment factors, and the error rate was reduced from 16.69 to 3% after applying the adjustment factors.

We also compared the R^2 value of before applying adjustment factors and the R^2 value of after applying adjustment factors; the R^2 was improved from 0.939 in Fig. 2 to 0.9927 in Fig. 3.

Next, we compared the PREDs of before applying adjustment factors and after applying adjustment factors, and the results of PRED were improved as shown in Table 10.

The high-performance models are often expected to have a high coefficient of determination R^2, small MMRE, and high PRED. Conte et al. [33] showed that the standard acceptance levels for an estimation model were PRED $(0.25) > 75\%$ or MMRE $< 25\%$. For the selected 19 software maintenance projects, we obtained $R^2 = 0.9927$, MMRE $= 3\%$, and PRED $(0.25) = 94\%$. Therefore, the proposed package software cost estimation model has a high performance and can be reliably applied to estimate the package software maintenance costs for actual projects.

4 Conclusion

Although the market for package software is growing rapidly, there are a few studies on estimating the software maintenance cost of package software. Therefore, this study investigated package software maintenance activities by reviewing previous research, classifying the identified activity elements, and proposed a cost estimation model based on the software maintenance activities. The activity-based model calculates the direct labor costs and estimates the total cost for software maintenance. To calculate the direct labor cost, the number of occurrences, the average service time required, the number of workers required, and the manpower level for each software maintenance activity are obtained.

In addition, we developed the adjustment factors to improve the accuracy of the model. Software age, Level of documentation, developer experience, enterprise maturity level, and level of software complexity was obtained as the adjustment factors.

To validate the model, we collected the cost data and data for adjustment factors for 19 actual package software maintenance projects from domestic companies, and compared the estimated cost obtained by the proposed model with the actual contract price for each project. The performance of the developed model was very high ($R^2 > 99\%$, MMRE $= 3\%$, and PRED $(0.25) = 94\%$).

This shows that the proposed model can be reliably applied to predict the package software maintenance costs for actual projects.

We developed a software package cost estimation model based on actual data. It is necessary to continuously collect and verify the actual cost data and adjustment factor data to improve the software package cost estimation model.

References

1. Gartner (2015), Gartner market databook update, Gartner, CT, USA, Connecticut
2. Lucia AD, Pompella E, Stefanucci S (2005) Assessing effort estimation models for corrective maintenance through empirical studies. Inf Softw Technol 47(1):3–15
3. Boehm BW, Horowitz E, Madachy R, Reifer D, Clark BK, Steece B, Brown AW, Chulani S, Abts C (2000) Software cost estimation with COCOMO II,Prentice Hall, NJ, USA
4. Juan CG, Manuel JB (1997) A method for estimating maintenance cost in a software project: a case study. Softw Maint Res Pract 9(3):161–175
5. Aggarwal K, Singh Y, Chandra P, Puri M (2005) Measurement of software maintainability using a fuzzy model. J Comput Sci 1(4):538–542
6. Pragya S, Varun KR (2012) A cost estimation of maintenance phase for component based. IOSR J Comput 1(3):1–8
7. Kim WJ, Jung SJ (2014) A method for estimating maintenance cost of package software, In Proc. of 4. International conference on computer science and information systems, Dubai, UAE, pp. 43–46
8. Boehm BW (1981) Software engineering economics. Prentice-Hall Inc, Englewood Cliffs
9. International standard ISO/IEC 14764 (2006) Software engineering-software life cycle processes-maintenance (ISO/IEC)
10. Musa JD, Iannino A, Okumoto K (1990) Software reliability: Measurement Prediction, Application. McGraw-Hill, Inc., New York
11. Hariza M, Voidrot JF, Minor E, Pofelski L, Blazy S (1992) Software maintenance: an analysis of industrial needs and constraints, In Proc. of the international conference on software maintenance, Orlando, Florida, pp. 9–12
12. Chapin N (1988), Software maintenance life cycle, In Proc. of the conference on software maintenance, IEEE computer society, CA, pp. 4–12
13. IEEE 1219 (1998), IEEE standard for software maintenance (IEEE)
14. IFPUG (2012) The IFPUG guide to IT and software measurement. CRC Press-Taylor and Francis Group, Boca Raton. https://doi.org/10.1201/b11884
15. ISBSG (2012), Managing your maintenance and support environment 2012 Update, ISBSG, Melbourne, Australia, South Melbourne
16. UKSMA (2001), Measuring software maintenance and support, Version 0.5, Draft, http://www.uksma.co.uk/,July 1st
17. Internet K, Agency S (2016) A guide for cost estimation of software project, 2016th edn. Korea Internet and Security Agency, Korea
18. Internet K, Agency S (2007) A guide for maintenance service of open source software. Korea Internet and Security Agency, Korea
19. Korea Internet and Security Agency (2010), An estimation for maintenance cost of information security systems in Korea industry Research Report 10-Policy-22, Korea Internet and Security Agency, Ministry of Knowledge Economy, Korea
20. Mather D (2005) The maintenance scorecard: creating strategic advantage. Industrial Press Inc., New York
21. Stark GE (1996), Measurements to managing software maintenance. In international conference on software maintenance (ICSM '96), pp. 152–161
22. Hunt B, Turner B, McRitchie K (2008) Software maintenance implications on cost and schedule, in Aerospace Conference IEEE, pp. 1–6
23. Korea Ministry of Information and Communication (2005) Package software maintenance service guide, Korea Ministry of Information and Communication
24. Dalkey NC (1969), The Delphi method: an experimental study of group opinion, Rand Corporation Memorandum RM-5888-PR
25. Shin HJ, Kim W, Jiro K (1998) The way of thinking of the information age, Sekyeong Multi Bank
26. Korea Ministry of Knowledge and Economy (2012) A guide for cost estimation of engineering projects, Korea Ministry of Knowledge and Economy, Notification 2012–178
27. Mehdi Hejazi Dehaghani (2013) Nafiseh Hajrahimi, Which factors affects software projects maintenance cost more? ACTA infrom MED 21(1):63–66
28. Banker RD, Datar SM, Kemerer CF (1987) Factors affecting software maintenance productivity: an exploratory study. In: Proceedings of the 8th international conference on information systems. ACM, New York, pp 160–175

29. Ogheneovo EE (2014) On the relationship between software complexity and maintenance costs. J Comput Commun 2(14):1–16
30. Alija N (2017) Justification of software maintenance costs. Int J Adv Res Comput Sci Softw Eng 7(3):15–23
31. Niessink F, Van Vliet H (1998) Two case studies in measuring software maintenance effort, 14th IEEE international conference on software maintenance (ICSM '98), pp. 76–85
32. Stark GE (1996) Measurements to managing software maintenance. International conference on software maintenance (ICSM '96), 1996, 152–161
33. Conte SD, Dunsmore HE, Shen VY (1986) Software engineering metrics and models. Benjamin/Cummings, Menlo Park