Analyzing Software Maintenance Cost Based on Work Efficiency and Unit Cost

Masateru Tsunoda
Nara Institute of Science and Technology
Nara, Japan
Department of Informatics, Kindai University
Osaka, Japan
tsunoda@info.kindai.ac.jp

Kenichi Matsumoto
Graduate School of Information Science
Nara Institute of Science and Technology
Nara, Japan
matumoto@is.naist.jp

Sawako Ohiwa
Economic Research Institute
Economic Research Association
Tokyo, Japan
er421@zai-keicho.or.jp

Tomoki Oshino
Economic Research Institute
Economic Research Association
Tokyo, Japan
er352@zai-keicho.or.jp

***Abstract*—Software maintenance is an important activity on the software lifecycle. In this study, we try to establish a benchmark of software maintenance cost. To establish the benchmark, factors affecting work efficiency and unit cost should be clarified, using a dataset collected from various organizations (cross-company dataset). We used dataset includes 837 data points collected by Economic Research Association from 2006 to 2016, and analyzed factors affecting work efficiency of software maintenance and unit cost of engineers. In the analysis, we defined two types of work efficiency. Also, as unit cost of engineers, we defined two types of unit cost. As attributes related to work efficiency, we analyzed address to process improvement, business sector, and required availability rate. As attributes related to unit cost, we analyzed address to process improvement, business sector, and social impact of faults. First, we showed correlation ratio of attributes to work efficiency and unit cost. Then, we analyzed each attributes using boxplots. As a result, business sectors related to both work efficiency and unit cost. The boxplots are useful to estimate software maintenance cost roughly.**

Keywords—cross-company dataset; unit cost; work efficiency; correlation ratio;

I. INTRODUCTION

Enterprise software needs software maintenance when a business process is changed. It often occurs, and hence users sometimes contract software maintenance with companies. Software maintenance does not mean only removing faults found after software release. Software needs extensions or modifications of its functions due to changes in a business environment, and software maintenance also indicates them. ISO/IEC 14764 [4] classifies software maintenance into followings:

- Corrective maintenance: modifications of faults found after software release.
- Preventive maintenance: corrective modifications before potential faults become actual faults, after software release.
- Adaptive maintenance: modifications to keep software availability against environmental changing after software release.
- Perfective maintenance: modifications for conservation or improvement of software performance or maintainability after software release.

It is important to establish a benchmark (reference values to compare an organization's performance with others [6]) of cost for software maintenance. For organizations which offer software maintenance service, the benchmarking is the basis of cost estimation and process improvement. The process improvement will enhance price competitiveness of the companies. For users (customer of software maintenance), the benchmarking is useful to evaluate the validity of the cost of the service supplier of software maintenance. If the cost of software maintenance is regarded to be higher than other service suppliers based on the benchmark, it gives the chance to reconsider the contract with the supplier.

In this study, we try to establish a benchmark of software maintenance cost. To establish the benchmark, we analyzed factors affecting work efficiency and unit cost of software maintenance engineers, using a dataset collected from various

978-1-5386-5605-1/18/$31.00 ©2018 IEEE
DOI 10.1109/BCD2018.2018.00024

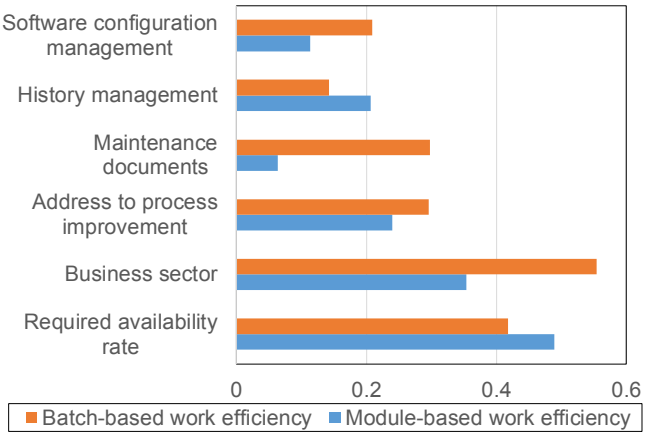

Fig. 1 Correlation ratio of work efficiency.

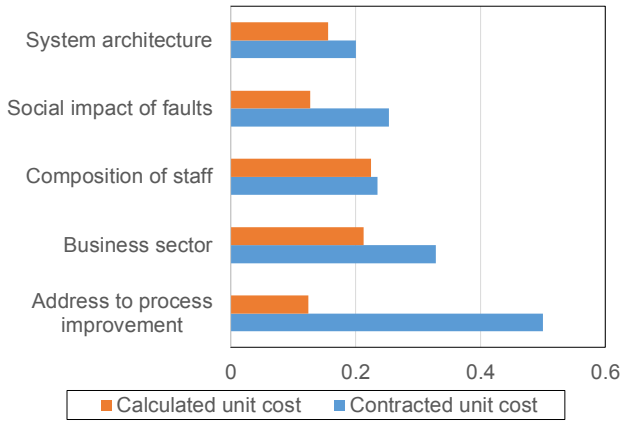

Fig. 2 Correlation ratio of unit cost.

organizations (cross-company dataset). The main contribution of our research is illustrating factors affecting work efficiency and the unit cost using cross-company dataset. It can be used to estimate software maintenance cost roughly. The work efficiency is defined as amount of software maintenance (i.e., the number of modified modules) divided by working time. Unit cost is cost of maintenance engineers per hour. The distributions of them are shown stratified by the factors. If one knows the amount of software maintenance, one can estimate total cost using the work efficiency and unit cost.

For example, the benchmarking shows median of work efficiency is 0.01 when business sector of the software is banking. If the number of modified modules is 10, working time will be 1000 hour (1000 = 10 / 0.01) for a year. When median of the unit cost of the engineers for banking system is $60, total cost is estimated as $60,000 roughly.

We previously analyzed work efficiency of software maintenance based on cross company dataset [10]. This study uses larger data, and analyzed the unit cost of engineers. In contrast, the previous study did not analyzed the unit cost. Also, we analyzed other factors such as process improvement activity. The factors were not analyzed in the previous study. Also, we analyzed factors related to work efficiency on software maintenance, using cross-company dataset [9]. In the study, work efficiency was defined as the number of modified modules per engineer. Therefore their definitions are rather rough. In contrast, this study and the above study [10] defines work efficiency based on working time. Although it is not easy for users to grasp working time for software maintenance, the definition is more precise. Note that the maintenance we focused on is only software maintenance, and it does not include system maintenance.

II. DATASET

The dataset used in the analysis includes 837 data points of the software maintenance agreement (project; for a year) which were collected from various organizations by the Economic Research Association. The yearly agreement was signed between 2006 and 2016 in Japan. So, the collected data such as the number of modified modules and working time were collected in a year. They send questionnaires to companies, and based on the responses, the dataset was made. Hence, we did not know how to record each attribute in detail. Note that generally, cross-company dataset is collected by similar way (e.g., Cross-company dataset [3] collected by ISBSG [2]).

To align the conditions of analyzed data and enhance the reliability of the analysis, we selected 107 data points which satisfy the followings:

- Total maintenance cost (contract price) in a year is recorded.
- Total maintenance cost is larger than 1,000,000 JPY (almost same as 10,000 USD).
- Total working time for software maintenance in a year is recorded.
- Ratio of maintenance activities classified by maintenance types (e.g., corrective maintenance is 20%) is recorded.

The work efficiency is defined as amount of software maintenance divided by working time. Although the amount of maintenance work is better to be measured by Function Point Analysis method, it includes many missing values. So, we used the number of modified modules and the number of modified batch files alternatively. This is because they are relatively correlated with modified Function Point in the preliminary analysis. Therefore, we defined two types of work efficiency as follows:

- Module-based work efficiency: the number of modified modules divided by working time in a year.
- Batch-based work efficiency: the number of modified batch files divided by working time in a year.

In the analysis, we used two types of unit cost of engineers as follows:

- Contracted unit cost: unit cost of staff which was answered in the questionnaire.
- Calculated unit cost: total cost of software maintenance divided by working time of engineers of

service supplier (i.e., it does not include working time of engineers of a user company).

To analyze the relationships between factors such as business sector and results such as work efficiency, we used correlation ratio. It is used to analyze a nominal variable and a numerical variable, and the range is from 0 to 1. The larger value denotes stronger relationship. In the analysis, we focused factors whose correlation ratio is larger than 0.2.

Fig. 1 shows correlation ratios of work efficiency and factors, and Fig. 2 shows that of unit cost and factors. In the next sections, we explain the analysis results of some relationships which are relatively strong in detail. In section III and IV, some outliers were omitted to enhance the visibility of the boxplots.

III. Analysis of Work Efficiency

A. Address to Process Improvement

Some companies try to improve software maintenance process to improve work efficiency and software quality. Such process improvement affects work efficiency. In the questionnaire, there are three types of address to process improvement as follows:

1. There is dedicated staff for process improvement.

2. There is concurrent staff for process improvement.

3. There is no staff for process improvement.

99. Other

Fig. 3 shows module-based work efficiency. When the value of the factor was 1 (i.e., there is dedicated staff for process improvement), position of the box was higher than the case where the value was 2 (i.e., there is concurrent staff for process improvement). Therefore, both the work efficiency of the top 25% (top of the box) and the bottom 25% (bottom of the box) were higher than 2, when the value was 1. The box size was large when the value of the factor was 3. This means variance of work efficiency is large, and the efficiency is very different in each case.

Fig.4 shows batch-based work efficiency. When the value of the factor was 1, work efficiency was low. This may be affected by the number of cases (it is only four). When the value was 2, median (horizontal line in the box) was larger than the case where the value was 3. So, the work efficiency of the former is larger. When the value of the factor was 3, the box size was large, and the efficiency was very different in each case. The analysis results show the followings:

- Setting process improvement staff may affect work efficiency.
- When there is no staff for process improvement, the efficiency is very different in each company.

B. Business Sector

Dataset used in this study was collected from various organizations, and therefore business sectors of organizations where maintained software works are also various. When the business sector is different, work efficiency of software maintenance may be also different. For example, when software whose business sector is banking is modified (i.e., maintained), testing software may take more time than other business sectors. This is because maintaining software may cause software fault, and the software needs rigorous testing to suppress the faults. As a result, amount of software maintenance per working hour (i.e., work efficiency) may be smaller than other business sector. For example, when a module of software for banking is modified, modified time is 8 hour and testing time is 4 hour. In contrast, when a module of software for other business sector is modified, modified time is 8 hour and testing time is 2 hour. In this example, work efficiency of former is lower than latter. Since the definition of work efficiency is modification time divided by working time.

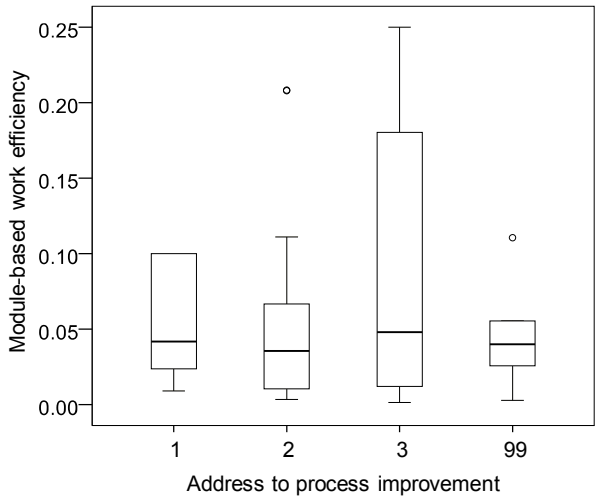

Fig. 3 Module-based work efficiency stratified by address to process improvement.

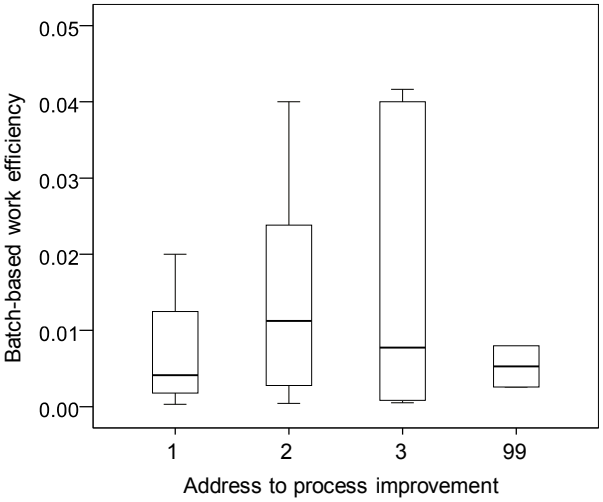

Fig. 4 Batch-based work efficiency stratified by address to process improvement.

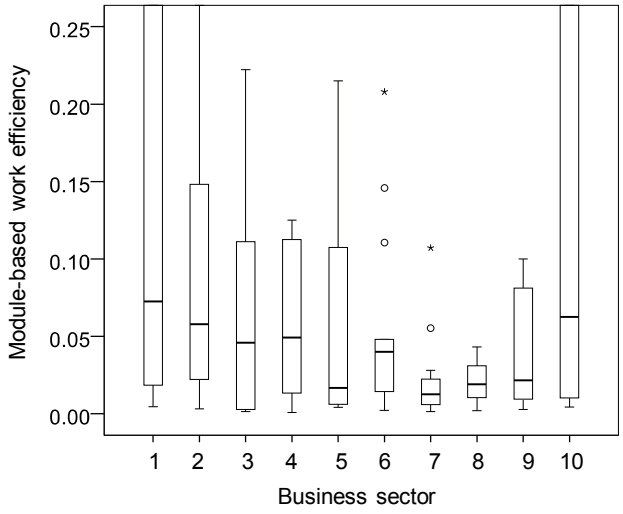

Fig. 5 Module-based work efficiency stratified by business sector.

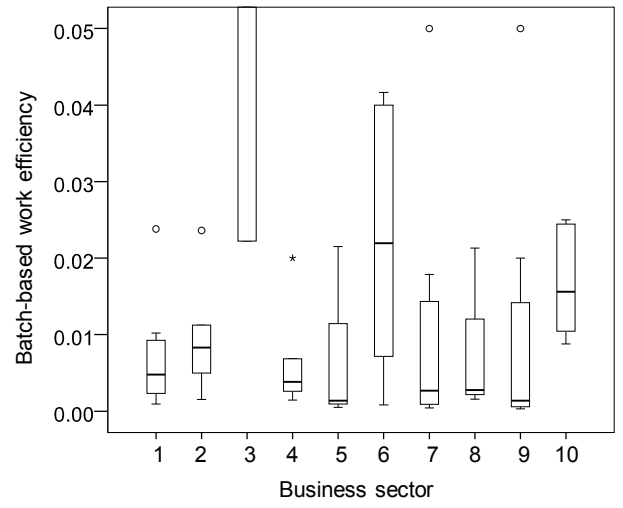

Fig. 6 Batch-based work efficiency stratified by business sector.

We focus on business sectors where the number of data point was not small (i.e., more than seven), and the tendency of module-based work efficiency and batch-based work efficiency was similar. Fig. 5 shows module-based work efficiency. In the figure, 2 (manufacturing) was higher, and 7 (banking and insurance) was lower. Fig. 6 shows batch-based work efficiency. In the figure, the median of work efficiency of 2 (manufacturing) was higher, and the variance of 7 (banking and insurance) was larger. When software development, developing software for manufacturing was higher work efficiency, and developing software for banking was lower work efficiency. So, when focusing on business sector, work efficiency of maintenance was similar to that of development. That is, maintenance of software for manufacturing is higher work efficiency, and maintenance of software for banking is lower work efficiency

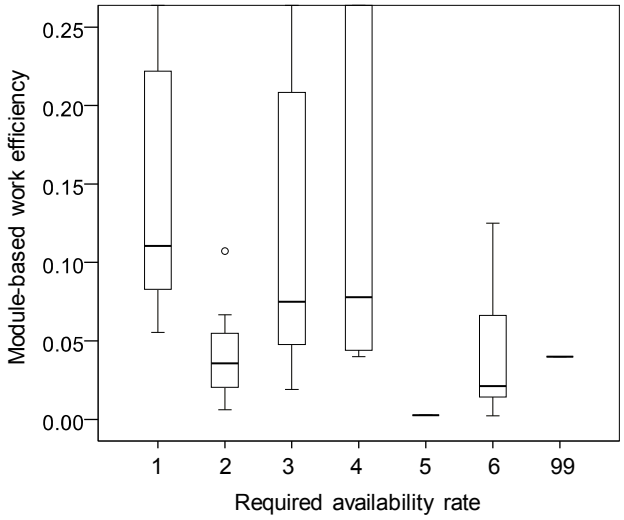

Fig. 7 Module-based work efficiency stratified by required availability rate.

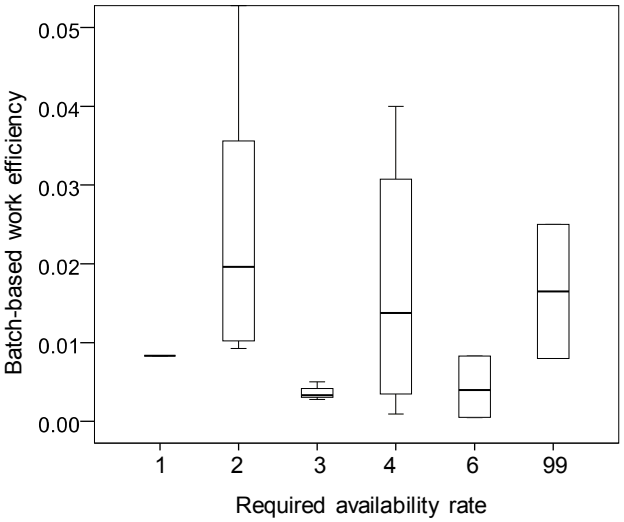

Fig. 8 Batch-based work efficiency stratified by required availability rate.

C. *Required Availability Rate*

Required level of availability rate is different for each software. Availability rate shows availability of software, and it is calculated as follows:

Availability rate = (total software operating time - total software unavailable time) / total operating time

The reason of the difference of the rate is that business which software supports is different for each organization, and when unavailability of software affects the business greatly, the required level is also high. When the required level is high, software testing may take more time to suppress software faults which are injected on software maintenance. This will lower work efficiency (see section III. B).

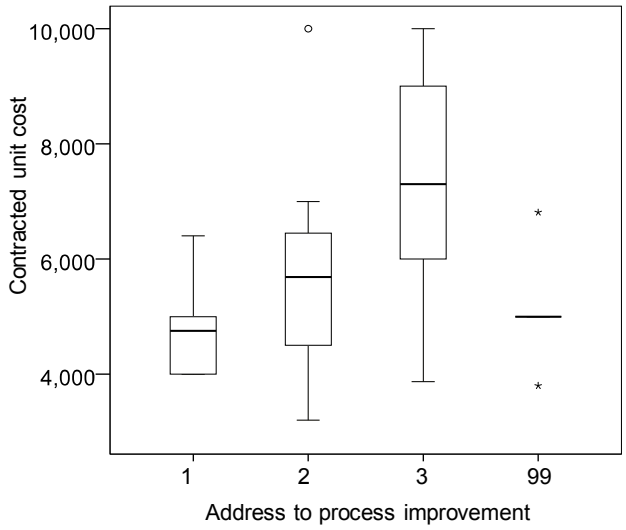

Fig. 9 Contracted unit cost stratified by address to process improvement.

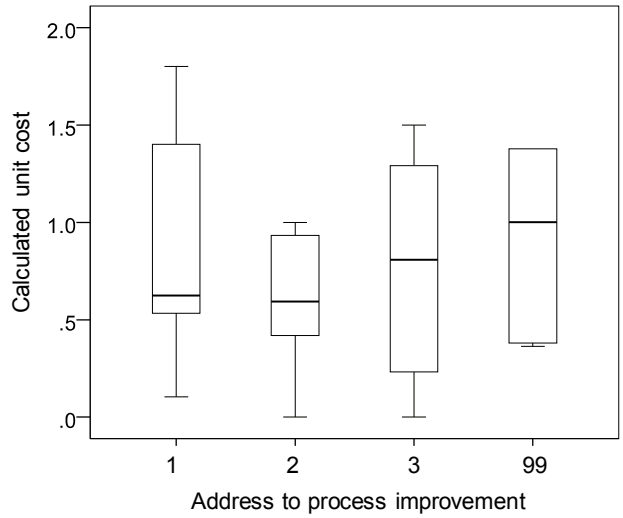

Fig. 10 Calculated unit cost stratified by address to process improvement.

In the questionnaire, there are six choices for required availability rate as follows:

1. 95% or below
2. Over 95% but no more than 99%
3. Over 99% but no more than 99.9%
4. Over 99.9% but no more than 99.99%
5. Over 99.99% but no more than 99.999%
6. Over 99.999%

99. Other

Fig. 7 shows the relationship between availability rate and module-based work efficiency. Compared to the required level 3 (over 99% but no more than 99.9%) and 4, the position of the box was lower when the required level was 6. We also show the boxplots of batch-based work efficiency in Fig. 8. Similarly, the position of the box of the level 6 was lower than the level 2 and 4. The result suggests that when required level of availability rate is high, work efficiency tends to be lower.

IV. Analysis of Unit Cost

A. Address to Process Improvement

We analyzed the relationship between unit cost of engineers and address to process improvement explained in section III. A. The difference of staff for process improvement is not considered to affect unit cost of engineers directly. This is because staff for process improvement may not support maintenance work directly. However, when an organization copes with process improvement (i.e., there is dedicated staff for process improvement), the process has been improved considerably, and there may be procedure and using for software maintenance. It may be possible that low cost (i.e., low skill) staff can maintain software by using the manual. So, we analyzed relationships between unit cost of engineers and address to process improvement.

Fig. 9 shows relationships of contracted unit cost. The number of data points of 1 (i.e., there is dedicated staff for process improvement) was small (the number is six). So, we focus on 2 and 3 on the figure. Median of 2 was smaller, and the position of the box (25 and 75 percentile of unit cost) was also smaller. Fig. 10 shows relationships of calculated unit cost. The number of data points of 1 was also small (the number is seven. Note that he number was different between Fig. 9 and 10 due to missing values). The median of 3 was the highest except for 99 (i.e., others). So, the result implies that unit cost of engineers is lower when there is staff for process improvement.

B. Business Sector

As explained in section III. B, there are various business sectors of organizations where maintained software is used. Maintenance tools and programming language are different for business sectors, and therefore required skills for engineers to maintain software are also different for that. It may be affected unit cost of engineers.

Boxplots of contracted unit cost are shown in Fig. 11, and that of calculated unit cost are shown in Fig. 12. We focus on business sectors where the number of data points is not small, and unit cost was similar tendency on both figures. The medians of constriction (2), information and telecommunications (4), and public services (10) were lower than other business sectors on both figures. In addition, there was no business sector whose median was higher than others on both figures. That is, unit cost of engineers is tends to be lower when business sector is constriction, public services, or information and telecommunications.

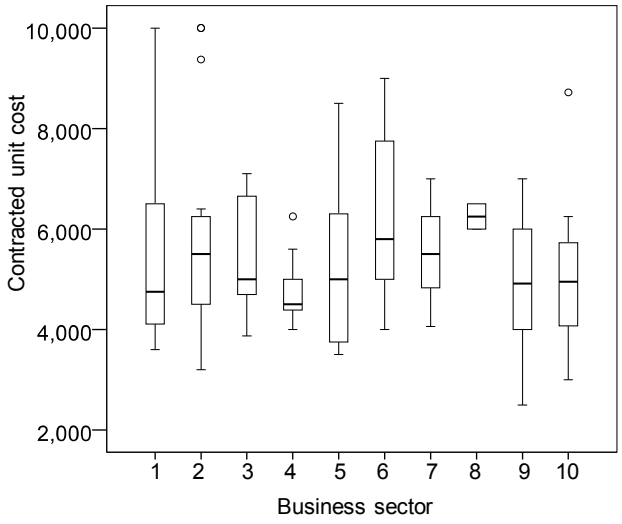

Fig. 11 Contracted unit cost stratified by business sector.

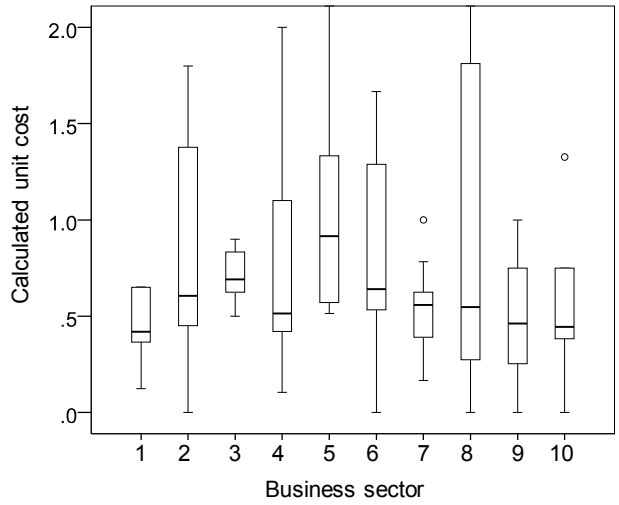

Fig. 12 Calculated unit cost stratified by business sector.

C. *Social Impact of Software Faults*

For some software, faults of the software affect social activities broadly. For example, when software is used as infrastructure of the society (e.g., traffic control system), the fault greatly affect social activities. When social impact of faults is high on software, high skill may is needed to maintain the software, and it makes unit cost of engineers higher. On the contrary, maintenance procedure may be explicitly defined on such software, and software is maintained based on the procedure. It may decrease unit cost of engineers because maintenance based on the procedure may not require high skill for engineers.

So, we analyzed relationship between social impact of software faults and unit cost of engineers. In the questionnaire, there are four choices for the social impact of maintained software as follows:

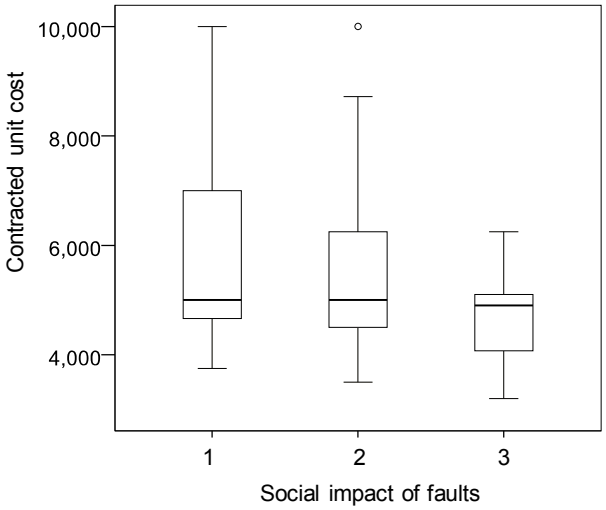

Fig. 13 Contracted unit cost stratified by social impact of faults.

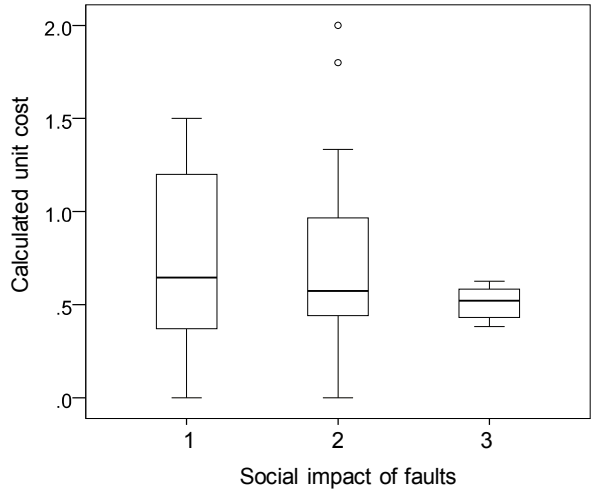

Fig. 14 Calculated unit cost stratified by social impact of faults.

1. Social impact of faults is almost nothing.
2. Social impact of faults is limited.
3. Social impact of faults is very large.

The number of data points where the answer to the item is 3 (i.e., social impact of faults is very large) was small. So, we focus on the other data points in the analysis. Boxplots of contracted unit cost stratified by the social impact is shown in Fig. 13. Although the median was almost same, top position (i.e., 25 percentile) of the box of 1 (i.e., social impact of faults is almost nothing) was higher than 2. Fig. 14 shows calculated unit cost. In the figure, the median and top position of the box were higher than 2. So, when social impact of faults is almost nothing, the unit cost is considered to be higher than others.

V. Related Work

Some studies analyzed work efficiency factors of software maintenance. Jørgensen [5] analyzed software company dataset, and showed that work efficiency is not affected by programming language. Ahn et al. [1] used variables which are similar to the productivity factors in a software maintenance effort estimation model. However, these researches did not analyze cross-company dataset.

ISBSG (International Software Benchmarking Standards Group) collects cross-company dataset of software maintenance [3]. However, it does not include price information such as unit cost of engineers. So, using the dataset, we cannot analyze price of software maintenance directly. Tsunoda et al. [11] analyzed the dataset, and they concluded that only several companies data can be used to analyze work efficiency on the dataset, due to missing values. In contrast, we analyzed data points collected from many companies (roughly speaking, each data point was collected from each company). So, the analysis results of this research are expected to have high generality.

There are few reports or researches which analyzed cross-company software maintenance dataset. Japan Users Association of Information Systems (JUAS) and Ministry of Economy, Trade and Industry used the cross-company dataset, and showed work efficiency stratified by business sector [7]. They defined maintenance cases per engineer as work efficiency, and their definition is a bit rough, compared with our definition. Also, JUAS reports the following results about software maintenance [8].

- Some rate of working time is used for communications about stakeholders.
- Requirement changes about software maintenance cause the delay of delivery time.
- Many companies conduct simultaneous modifications for reducing workload of software maintenance.
- Many companies adopt engineer awards programs for generating motivations.

Note that the results are based on the summery of the questionnaire. That is, they did not analyze the relationships between work efficiency and the factors quantitatively in the report [8].

VI. Conclusions

In this study, we tried to establish a benchmark of software maintenance cost. We used the dataset collected from many companies, and analyzed attributes related to the work efficiency and unit cost. The analysis results show that some attributes are related to work efficiency and unit cost as follows:

- Setting process improvement staff may affect work efficiency.
- Maintenance of software for manufacturing is higher work efficiency, and the maintenance for banking is lower work efficiency.
- When required level of availability rate is high, work efficiency tends to be lower.
- Unit cost of engineers is lower when there is staff for process improvement.
- Unit cost of engineers is tends to be lower when business sector is constriction, public services, or information and telecommunications.
- When social impact of faults is almost nothing, the unit cost is considered to be higher than others.

The analysis results are useful for users and service suppliers to benchmark their cost using boxplots shown in this study. Note that the benchmarking should be used as reference, but not as rigid criteria. Since the variance of work efficiency and unit cost is not small.

Acknowledgment

This research was partially supported by Japan Society for the Promotion of Science(JSPS) [Grants-in-Aid for Scientific Research (C) and (A) (No.16K00113 and No.17H00731)].

References

[1] Y. Ahn, J. Suh, S. Kim, and H. Kim, "The software maintenance project effort estimation model based on function points," Journal of Software Maintenance: Research and Practice, Vol. 15, No. 2, pp. 71-85, 2003.

[2] International Software Benchmarking Standards Group, https://www.isbsg.org/

[3] International Software Benchmarking Standards Group (ISBSG), ISBSG Maintenance & Support Data Suite Release 4, ISBSG, 2010.

[4] ISO/IEC 14764:2006, Software Engineering - Software Life Cycle Processes - Maintenance, International Organization for Standardization, 2006.

[5] M. Jørgensen, "Experience With the Accuracy of Software Maintenance Task Effort Prediction Models," IEEE Transactions on Software Engineering, Vol. 21, No. 8, pp. 674-681, 1995.

[6] C. Lokan, T. Wright, P. Hill, and M. Stringer, "Organizational Benchmarking Using the ISBSG Data Repository," IEEE Software, Vol. 18, No. 5, pp. 26-32, 2001.

[7] Japan Users Association of Information Systems (JUAS), Software Metric Survey, JUAS, 2008 (In Japanese).

[8] Japan Users Association of Information Systems (JUAS), Software Metric Survey 2013, JUAS, 2013 (In Japanese).

[9] M. Tsunoda, A. Monden, K. Matsumoto, and T. Oshino, "Analysis of Software Maintenance Efficiency Focused on Process Standardization," Proc. of International Workshop on Empirical Software Engineering in Practice (IWESEP), pp.3-8, Nara, Japan, 2011.

[10] M. Tsunoda, A. Monden, K. Matsumoto, S. Ohiwa, and T. Oshino, "Benchmarking Software Maintenance Based on Working Time," Proc. of Applied Computing and Information Technology (ACIT), pp.21-28, Okayama, Japan, 2015.

[11] M. Tsunoda, and K. Ono, "Pitfalls of Analyzing a Cross-company Dataset of Software Maintenance and Support," Proc. of International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp.349-354, Las Vegas, NV, USA, 2014.