Software Maintenance Productivity and Maturity

Jean-Marc Desharnais
TAM and Softlab
Bogaziçi University
34342 Bebek, Istanbul, Turkey
+90 216 516 3537
desharnaisjm@gmail.com

Alain April
Software Engineering Department
École de Technologie Supérieure
Montréal, Canada H3C 1K3
+1 514 396 8682
alain.april@etsmtl.ca

ABSTRACT

Maturity models assess the organization's processes to determine their level of maturity and capability. There is an implicit assumption that a higher level of maturity (or more capability) leads to higher level of productivity and quality. Based on this assumption, maintenance organizations that implement a number of exemplary practices should show an improvement in both. In this article, we use data from a maintenance organization to verify this assumption. The introduction presents the challenges associated with the measurement of software maintenance productivity and quality. We then introduce our methodology, followed by an analysis of the data, and, finally, lessons learned and future work.

Keywords

Software maintenance, productivity measures, S3m, maturity model

1. INTRODUCTION

A process maturity model is "a process improvement approach that provides organizations with the essential elements of effective processes that ultimately improve their performance."[1] The SEI CMMI (Capability Maturity Model Integrated) is a well-known process model, which is used with its corresponding process assessment guide (SCAMPI). While the CMMI mainly addresses software development projects (and most recently acquisitions and services), more specific models are proposed for small software maintenance that "address the assessment and improvement of the software maintenance function by proposing improvements to the software maintenance organizations and introducing a proposed maturity model for daily software maintenance activities: Software Maintenance Maturity Model (S3M)"[2]. The assessment process at the Software Engineering Institute (SEI) is called SCAMPI (Standard CMMI Appraisal Method for Process Improvement)[3]. A scaled-down version of SCAMPI was developed for small maintenance teams and used with S3M) [9]. There is an implicit assumption in the use of CMMI and S3M: if the recommended exemplary practices are deployed across the organization, quality and productivity will improve.

Measuring the quality and productivity of software maintenance within an organization is not an easy task. Many organizations have a number of individuals dedicated to maintaining their software applications. The productivity of software maintenance is difficult to assess because the resources needed for maintenance do not necessarily relate to the amount of work to be done. Improvement of a specific software application depends of the resources available, which means that work requests with higher priorities will be addressed first. The productivity data must be interpreted carefully; for example, if, during the summer months, the effort expended on maintenance is lower than it is during the other months of the year, it is probably because individuals are on vacation, not because there is an increase in productivity.

In contrast, the quality of a software application is rarely only related to the quality of the maintenance process, but also to the development process. Poor development leads to more defects, failures, and badly developed software, all of which have an effect on the resulting software maintenance effort. A software application could be of low quality, but this does not mean that the maintenance group has done a poor job. Productivity and quality results need to be interpreted cautiously, and this is generally difficult at the software application level if the data are only collected over short periods of time. Trends tend to reveal themselves over time.

2. METHODOLOGY

This article presents measures adopted by a specific maintenance organization with more than 200 employees. These are measures that use few attributes. The organization had been working to improve many of their processes, both development and maintenance, and already had numerous measures in place which could be used to evaluate productivity and quality. Their main measurement objective was to assess the effect of improvement on maintenance processes and the maintenance budgeting process.

The measures used in this case study are size, effort, and number of work requests by maintenance category:

- Functional size: The functional sizes of the software in the application portfolio were measured. The functional size measure chosen was (unadjusted) Function Point Analysis (FPA)[4]. Functional size is necessary in productivity calculations, because it varies across applications and is required for assessing relative effort (hours per 1000 FP for a period of time) when comparing the productivity of two software applications.
- Effort required to complete a maintenance request: The organization has a computerized time reporting system which keeps track of the identification number of each maintenance request, the application concerned, the actual effort (person-days) used to complete the maintenance requests, the maintenance category, etc. This maintenance request tracking system was therefore used to obtain the effort per maintenance request.

Table 1. Maintenance Categories.

Category	Description
Adaptive	Modifications to adapt a software product to changes in data requirements and processing environments [5]
Corrective	Reactive modification of a software product performed after delivery to correct the faults

	discovered [7] Modification repair code to satisfy functional requirements
Preventive	Modification of a software product after delivery to detect and correct latent faults before they become operational faults [7]
Perfective	Modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures [7]
User support	Response to user demands other than adaptive, corrective, preventive, or perfective [5]

- Maintenance categories: Classifying maintenance requests in accordance with the terminology recommended by international standards could be a challenging task. In this specific case, the measurers adapted the organization maintenance categories to the five categories recognized by the international standard ISO/IEC 14764. Using these categories always helps in benchmarking and future research [5]. These are similar to Pigoski's categories [6].

To obtain environmental characteristics, a questionnaire was created. We asked senior maintainers in charge of each software application to be measured to complete a questionnaire requesting information on:

- Application identification and description
- Technical constraints (response time, security, number of users, platforms)
- Maintenance tools and techniques used (development methodology, CASE tools)

The measures and the questionnaire responses were used to obtain details about each software application and to help explain quality and productivity differences, if we were to find any. Although we captured environmental characteristics for each software application, they were not used in this research. We concluded that the organization's software applications were very homogeneous, and that this factor had little or no influence on the final results. Some of the data used to come to this conclusion are presented in Appendix 1.

3. DATA PRESENTATION AND ANALYSIS

3.1 Category analysis

The results presented in this study are presented by application and by maintenance category (for all applications).

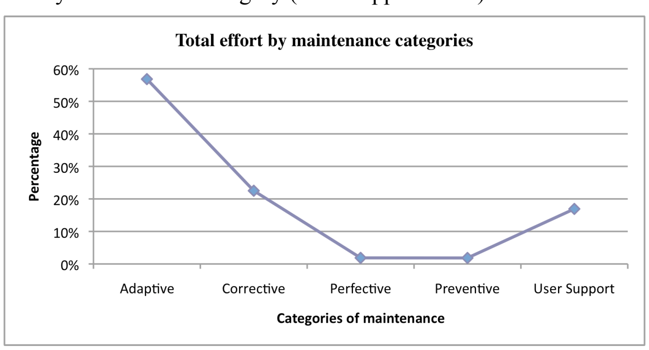

Fig. 1. Total effort by maintenance categories

We present intervals of: one-year, three-month for a total of 39 consecutive months measured. Figure 1 shows the relative percentage of effort by maintenance category for 39 consecutive months (14 periods). We observed that the adaptive and corrective maintenance categories accounted for 80% of the total effort, perfective and preventive just 3%, and user support 17%.

Figure 2 shows the functional size of each application over the years. There is relatively little variation, which means that the size variations of specific applications will not have very much affect on productivity.

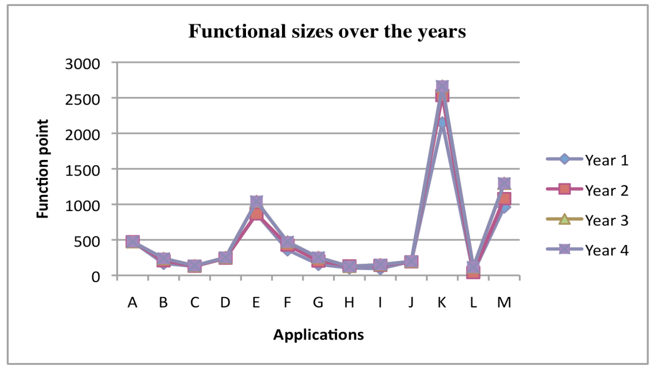

Fig. 2. Functional sizes over the years per application

Figure 3 shows the relative effort as a percentage of user support.

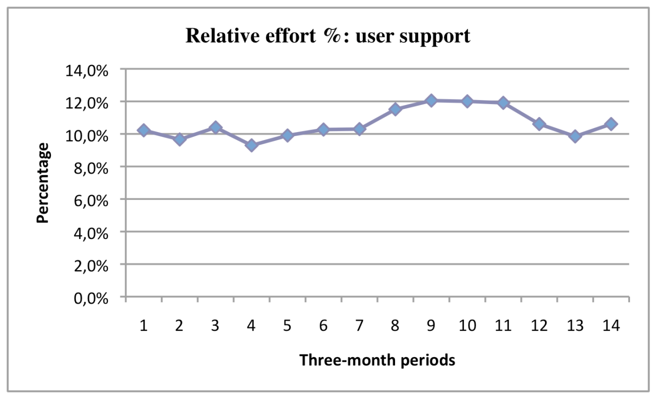

Fig. 3. Relative effort as a percentage (user support)

This shows that the variation in relative effort per three-month period for user support is low (between 9.3% and 12.1%). The maintenance manager's explanation was that the user support personnel were generally not used tasked with adaptive or corrective maintenance, as they had different expertise. It is important to note that the number of maintainers remained stable over the year. In comparison, the relative effort as a percentage for corrective maintenance varies between 18% and 42%, and the relative effort as a percentage for adaptive maintenance varies between 41% and 67%. The same maintainer can perform corrective or adaptive maintenance, depending on the priorities set. Finally, because perfective and preventive maintenance were not representative of the global efforts, and user support work is performed by a stable, independent group (see Figure 3), we concentrated our analysis on the adaptive and corrective maintenance categories.

3.2 Adaptive and corrective maintenance analysis

Figure 4 shows the variation in functional size of each application over the years. The variation in functional size between the applications is high, while the variation within each application over the years is low. In this application portfolio, the smallest size found for an application was 40 FP and the largest was 2530 FP. The variation between the largest and the smallest functional size was 63. The minimum variation of the size of an application over the years was 1.39, and the maximum was 2.89. For our analysis, considering the size of an application is the important factor, while the variation over the year is less important.

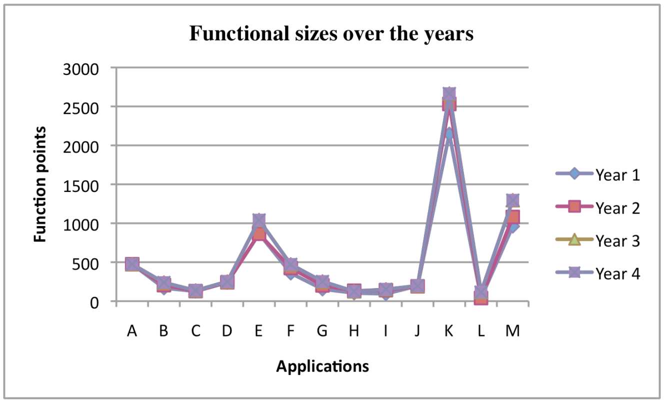

Fig. 4. Functional sizes over the years.

Figure 5 shows the average effort per 100 Function points by application for adaptive and corrective maintenance. We observed that, on average, application F shows a high relative cost (days per 100 FP per year) for adaptive maintenance, while corrective maintenance cost 20 days per 100 FP, which is close to the 17.4 average relative cost for corrective maintenance over the years. The adaptive maintenance cost for application B is also low (27), but the corrective maintenance cost (at 42) more than twice the average relative cost of corrective maintenance.

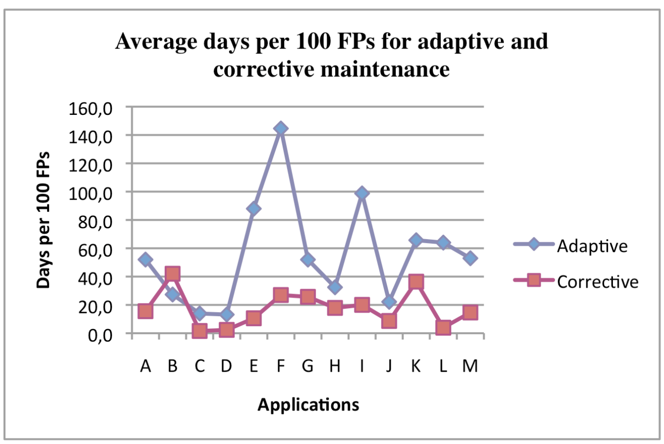

Fig. 5. Average days per 100 FPs by application for adaptive and corrective maintenance.

A closer analysis of the relative cost for adaptive and corrective maintenance for applications B and F showed the following results (Figure 6). Application F shows a higher relative cost for adaptive maintenance compared to application B, but this varies over the years. The relative cost (days) for application F is lower than for application B in terms of corrective maintenance (except for the last year measured). However, we should be careful when interpreting the fourth year figure (*), because only 3 months of data were collected in that year. We observe that there are variations over the years that should be explained application by application.

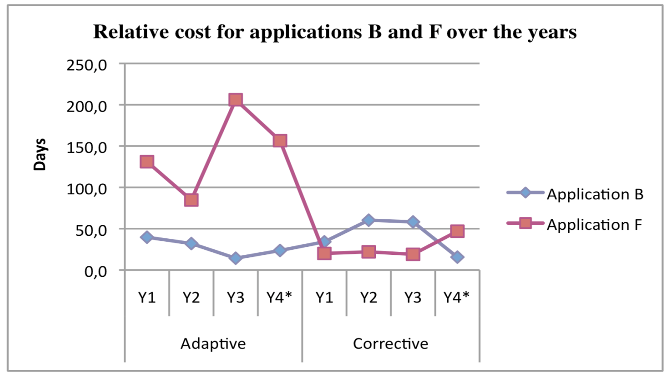

Fig. 6. Relative cost (days) for applications B and F over the years.

For example, the corrective maintenance for application B is higher for years 2 and 3 and drops for year 4. Why is that? This example shows that interpreting productivity and quality for a specific application over the years is not a trivial task.

Figure 7 shows the percentage of adaptive and corrective maintenance over the years for all applications. Over the years, the percentage of effort increased for adaptive maintenance and decreased for corrective maintenance. Figure 7 shows the relative time expended each year to correct or adapt functionalities on overall applications after measuring the effort by category and the size of those applications. From our definitions, corrective maintenance is essentially correction of failures and defects (which means that there is no new functionality for the user in this maintenance category). Adaptive maintenance occurs when the user asks for new functionalities or better processing control of the existing application.

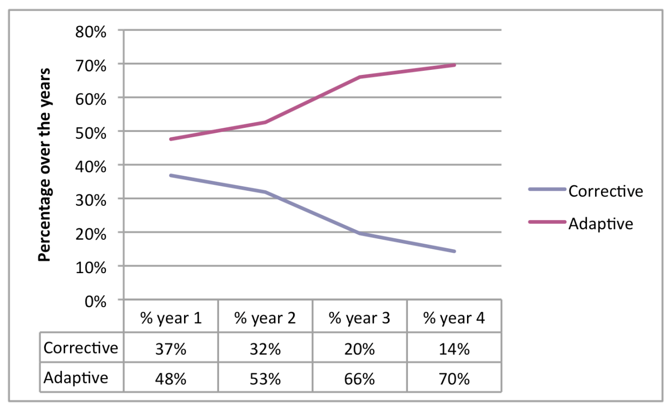

Fig. 7. Percentage of adaptive and corrective maintenance over the years.

Using the data for all the applications shows a tendency which can be interpreted more easily knowing the context of the organization. Figure 7 might be an indication that the efforts to improve the development and maintenance processes may result in an increase in productivity and quality.

4. LESSONS LEARNED AND FUTURE WORK

A number of lessons were learned using data to interpret maintenance productivity and quality:

- The effort by category of maintenance was observed mainly on corrective and adaptive maintenance. The efforts expended on perfective and preventive maintenance are around 2% of the total effort for each of them. It will be interesting to discover how an increase in effort in those categories can change productivity and quality. In this context, making a distinction in adaptive maintenance between adding new functionalities and adding better control and quality could be interesting, because of the significant effort expended in the adaptive category;
- The functional size of maintained applications changed slightly over the years. The increase is 39% on average. However, the final size of an application can be misleading, because the deletion and replacement of functionalities are also part of change requests;
- There is a substantial difference in functional size between the maintained applications. The largest application is 63 times the functional size of the smallest one. If the difference in productivity were known, the difference in cost of maintaining a larger application versus a smaller one could be analyzed;
- The relative cost variation by application maintained (corrective and adaptive maintenance) over the years is not linear. At the level of a specific application, there are variations, and statistical analysis becomes more difficult to use in this case. The relative cost of user support over the years for all applications is stable, but varies from one application to another;
- In this case study, we found a relation between process improvement and productivity and quality.

This type of analysis should be undertaken in different organizations to confirm the lessons learned. As well, four years is a relatively short time in the maintenance life of an application, and analysis should be performed over a longer period. The International Software Benchmarking Standards Group (ISBSG) [8] also suggests looking at maintenance based on categories similar to those used in this article. However, as far as we know, no analysis of this kind has been performed over a period of years for a specific organization.

APPENDIX 1: DATA SAMPLE

	Cost					
	App.	Adaptive	Corrective	Perfective	Preventive	User Support
Year 1	A	55.5	16.7	1.4	0.0	0.5
	B	39.7	34.1	0.0	0.0	5.4
	C	4.7	2.8	0.2	1.3	9.2
	D	18.5	3.3	0.8	0.6	7.6
	E	93.6	8.3	3.1	1.0	14.5
	F	131.1	20.1	6.8	9.9	20.4
	G	66.4	13.3	1.4	25.9	16.9
	H	41.6	20.8	1.4	7.9	21.0
	I	245.8	33.1	2.8	4.4	49.0
	J	18.8	6.4	0.9	2.0	11.1
	K	11.8	62.9	1.1	0.2	39.2
	L	100.5	1.0	1.0	0.0	12.5
	M	66.7	12.9	10.2	0.5	19.2
Year 2	A	57.5	26.1	15.9	0.0	0.5
	B	32.0	60.3	0.0	0.0	22.2
	C	14.0	1.1	0.6	0.1	4.6
	D	17.2	2.5	0.2	2.5	2.5
	E	82.0	6.5	0.4	0.3	13.2
	F	84.7	22.0	0.8	3.3	15.8
	G	84.5	55.8	0.0	0.0	12.3
	H	19.8	21.8	0.1	0.0	8.0
	I	106.1	23.1	3.0	0.0	16.5
	J	5.1	8.3	0.0	0.0	5.7
	K	38.8	24.7	0.6	2.1	37.7
	L	82.0	9.8	0.0	0.0	14.1
	M	47.4	26.4	6.8	0.7	8.7
Year 3	A	49.2	11.3	0.4	0.0	7.9
	B	14.2	58.2	2.1	0.0	12.7
	C	20.7	1.1	0.0	0.0	7.7
	D	9.5	1.6	0.4	0.0	3.5
	E	72.8	5.8	0.7	0.2	9.0
	F	206.1	19.0	2.2	0.0	19.7
	G	35.6	10.1	0.0	0.0	7.3
	H	47.1	6.1	4.4	0.3	12.1
	I	27.8	9.2	0.0	1.2	16.4
	J	23.7	12.3	0.0	0.0	9.2
	K	96.2	38.8	0.4	6.0	18.3
	L	73.5	4.8	0.0	0.0	2.9
	M	40.5	11.9	0.3	2.8	7.3
Year 4*	A	11.5	2.0	0.0	0.0	1.6
	B	5.9	3.9	4.7	0.0	2.9
	C	4.1	0.4	0.0	0.0	1.0
	D	1.9	0.5	0.0	0.0	0.9
	E	25.9	5.4	0.7	0.4	3.2
	F	39.1	11.7	2.8	0.0	7.5
	G	5.4	5.9	0.0	0.0	3.5
	H	5.3	5.7	0.0	0.0	3.4
	I	3.8	3.7	0.0	0.0	3.4
	J	10.3	1.9	0.0	0.0	4.2
	K	29.0	4.8	0.0	1.3	4.9
	L	0.0	0.0	0.0	0.0	1.6
	M	14.3	1.9	0.0	0.1	3.1

(*) Year 4 only 3 months

5. REFERENCES

[1] Capability Maturity Model Integration for Development (CMMi-Dev), Version 1.2, CMU/SEI-2006-TR-008, ESC-TR-2006-008. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, (2006) 561 p. : http://www.sei.cmu.edu/reports/06tr008.pdf

[2] April A., Abran A.: Software Maintenance Management: Evaluation and Continuous Improvement in Wiley-IEEE, (2008) 314 p.

[3] CMU/SEI (2006). CMU/SEI-2006-TR-011: Appraisal Requirements for CMMI, Version 1.2 (ARC, V1.2), SCAMPI Upgrade Team. Pittsburgh, PA, Carnegie Mellon Software Engineering Institute.

[4] IFPUG FP CPM: International Function Point Users Group (IFPUG) Function Point Counting Practices Manual, (2000) Release 4.1.1

[5] Abran A., Nguyenkim H.: Measurement of the Maintenance Process from a Demand-Based Perspective in Journal of Software Maintenance: Research and Practice 5: 2. (1993) 63-90

[6] Pigoski T. M.: Practical Software Maintenance: Best Practices for Managing your Software Investment in Wiley, (1997) 384p.

[7] International Organisation for Standardization. Software Engineering – Software Life Cycle Processes -- Maintenance, ISO/IEC Standard 14764. International Organisation for Standardization: Geneva, Switzerland, (2006)

[8] International Software Benchmarking Standards group - ISBSG. Data Collection Questionnaire: Application Software Maintenance and Support. Version 2.3, International Software Benchmarking Standards group: Victoria, Australia, (2010) http://www.isbsg.org/

[9] Lebrun, V., April, A.: Evaluation method based on S3m, University of Namur, (2007), 35p. http://www.s3m.ca/