An Empirical Study of Maintenance Issues within Process Improvement Programmes in the Software Industry

Tracy Hall, Austen Rainer, Nathan Baddoo, Sarah Beecham
Department of Computer Science, University of Hertfordshire, UK
{t.hall, a.w.rainer, n.baddoo, s.beacham}@herts.ac.uk

Abstract

Anecdotal evidence from our work with software developers suggests that maintenance is a significant problem for software development companies. A problem that is absorbing increasing amounts of precious development effort. In parallel, software companies are increasingly applying process improvement principles to development problems. In this paper we discuss how maintenance is addressed in process improvement programmes. We look at how well maintenance is addressed by formal process models like CMM. We also present empirical evidence from our study of process improvement in UK software companies. Our main findings are that although developers report that maintenance is indeed a problem, it is not always their most important problem. Furthermore, our findings also suggest that companies are often not well prepared for the maintenance phase of developments and that formal process improvement models do not pay enough attention to maintenance.

1. Introduction

In this paper we look at how process improvement programmes address software maintenance. Anecdotal evidence suggests that maintenance is a major problem for companies and we look at how companies are applying process improvement to reducing their maintenance burden. We present empirical evidence that characterises the process improvement efforts of 85 companies. Our results suggest that although companies are certainly aware of the maintenance burden created by fault-ridden software that does not satisfy users, companies are generally poor at targeting process improvement at the reduction of maintenance effort. Furthermore process models like CMM and ISO9001 have been criticised for not addressing maintenance activities directly enough [5]. Indeed even influential CMM publications [13] contain no references to maintenance in their index. Such publications also seem to consider maintenance as a separate issue to development as illustrated in the following quote (though the sentiment in the quote is well intended):

> *"…in both development and maintenance it is desirable to trace bugs to find the specific defects and errors that caused them"* [11, p312]

In this paper we present qualitative evidence from software developers in three case study companies to reveal the scale of maintenance problems existing in their companies. We found that many of the development problems that developers reported to us are directly related to maintenance. Developers report major problems in companies with faults delivered to customers coupled with major problems with low user satisfaction levels. Indeed we found that poor requirements capture contributes many problems to maintenance.

We followed up these case study findings by analysing questionnaire responses from 85 companies to determine how these companies are addressing maintenance issues within process improvement. We found that many companies are not effectively incorporating maintenance factors into process improvement activities. Furthermore very few companies report that they monitor maintenance activities, which suggests that many companies have an inadequate understanding of their maintenance activities.

The work we present here is part of our wider empirical study of process improvement in software companies [3, 9]. The data we present was collected during our study of process improvement in thirteen case study companies and the follow-up survey of process improvement in 85 companies. This means that our findings are based on a combination of qualitative and quantitative data. Overall our findings suggest that although companies are not effectively focussing on reducing maintenance effort, a few changes of emphasis could dramatically improve the situation in many companies.

In section two of the paper we explain our study methods and describe the companies in the study. In section three we present our findings, including qualitative data characterising the maintenance features of the case study companies and quantitative survey data describing how maintenance is being tackled in process improvement programmes. We further discuss our findings in section four, present ideas for future work in section five and conclude in section six.

2. The study methods

2.1. Company Case studies

We conducted detailed case studies in thirteen software companies. At each of these companies we held focus group sessions with three types of staff group: senior managers, project managers and developers. Membership of each group consisted of between four and six people. Overall we conducted 44 focus groups which involved approximately 200 software professionals. During focus groups we encouraged participants to discuss various topics associated with software process improvement. One such topic related to the problems companies had in their software processes. It is the results from these discussions we report in section 3.1.

Subjective data elicited from software professionals in this way is unusual in software engineering research and yet, where this approach has been used in other studies highly accurate data has been collected [1].

2.2. Questionnaire

We followed-up our case studies with detailed questionnaires to a wide range of additional software companies. We received completed questionnaires from 85 software companies. In the main, process improvement managers or quality managers completed the questionnaires on behalf of companies. Again the main aim of the questionnaire was to gather quantitative data characterising process improvement efforts in companies, however, the results contain a great deal that is relevant to maintenance. Results related to maintenance are presented in section 3.2 of this paper.

The questionnaire was designed according to classical questionnaire design principles [10] and can be viewed at [http://homepages.feis.herts.ac.uk/~pppgroup/questionnaire.html].

2.3. Companies in the study

The tables in Appendix One and Appendix Two provide some basic demographic data characterising the companies in the study. The tables show that a range of companies is represented in both our case study companies and in our questionnaire sample. Appendix Two shows that the questionnaire sample of companies has the following characteristics:

- A balance of UK and multi-national companies.
- A range of software development function sizes.
- A varied profile of company ages, with most companies having been established over ten years ago and many over twenty.
- A very high proportion of ISO9001 certified companies. This is probably a reflection of our targeting techniques rather than being representative of the industry as a whole. A few companies in the sample have been CMM assessed but not many.
- A balanced representation of application areas, also with a well balanced split

between bespoke and commercial development activities.

- A broad focus of software development effort, with development, maintenance, support and operations complementing a small amount of consultancy effort.

3. Findings

3.1. Case Study Findings

In this sub-section we present our provisional analysis of the data we collected from our 44 focus group sessions. We have not yet analysed all of the data collected and therefore we present data from three companies, companies 2, 5 and 10 (see Appendix Two). The data we present was collected from thirteen focus groups.

In each focus group we asked practitioners at each of three hierarchical levels what they considered were the most important problems and issues in the software process. In our provisional analysis we categorised the problems cited into three major categories: organisation problems, project management problems and development problems. Each category was refined to another level of detail, for example the development category was refined to problems associated with: requirements, design, coding, testing and maintenance. We then classified each issue that arose and counted the occurrence of issues in each sub-category. Table 1 shows the spread of problems across the major categories. It shows that practitioners at all levels in the three companies consider that organisational issues contribute almost half of all problems to software development. Table 1 suggests that development problems only contribute a quarter of all problems experienced in the software process.

Table 1. Problems cited in focus groups

Problems cited in 13 focus groups

	Frequency	Percentage
Organisational issues	141	47
Project management issues	80	27
Development issues	77	26
Total number of problems	298	100

Table 2 shows how problems experienced within development are broken down into particular lifecycle phases. It shows that of all the development problems cited only 18% were attributed directly to maintenance activities. This means that of all the problems identified by practitioners only 6% of them were directly related to maintenance.

Table 2. Development problems cited in focus groups

Development problems cited in 13 focus groups

	Frequency	Percentage
Requirements	31	40
Design	7	9
Coding	6	8
Testing	19	25
Maintenance	14	18
Total number of lifecycle problems	77	100

These results imply that development issues constitute a minority of problems experienced by practitioners and that even within development issues, maintenance activities generates fewer problems than requirements capture and testing. Requirements particularly cause most problems in the software process, indeed the following quote from a software developer in one of the case study companies aptly illustrates the requirements problems that are encountered:

> *"It is possible for us to start a project, get half way through it and the customer will turn around and say, this is now going to be used in a safety critical application…"*

Closer analysis of the results in Table 2 suggest that the vast majority of development problems impact significantly on maintenance. Many requirements problems directly affect maintenance and similarly design, code and test problems. Indeed we speculate that Table 2 identifies the degree to which other aspects of the development process contributes to maintenance effort. It shows that requirements problems contribute significant problems to maintenance. Table 2 further suggests that actually performing maintenance tasks does not present too many problems. Performing design and coding activities creates least problems.

3.2. Questionnaire Findings

In this section we present results from a questionnaire completed by 85 software companies. The questionnaire is intended to follow up and examine in a more quantified way the qualitative data we collected from our case study companies. In many of our case study companies practitioners told us that delivering faults to customers and repairing those faults was a major headache for them and their managers. Practitioners at all levels expressed concern about this. Practitioners in many companies told us that reducing such faults and controlling maintenance effort were important business objectives. To address such objectives companies need to do two things:

- Ensure that objectives are explicit.
- Collect measurement data tracking progress towards these objectives.

Without explicit goals and objectives it is difficult for companies to focus on improving specific areas, like reducing maintenance effort, and without measurement data it is impossible to monitor trends towards the achievement of objectives [14, 8]. In this sub-section we discuss how companies in our sample of 85 companies addressed the goals and measurement data that relate to maintenance.

3.2.1. Maintenance as an improvement criteria

Many practitioners in our 13 case study companies told us that reducing maintenance effort was an important business objective of their company. In particular, practitioners told us that they were under pressure to reduce delivered faults and to become more efficient at maintenance tasks. We used our questionnaire of 85 companies to find out how companies were addressing these issues in their software process improvement programmes.

We first asked respondents (process improvement managers) to tell us why process improvement was in place in their company. Although 67% of respondents said that improving product quality was the main reason, no respondents said that process improvement was directed at any specific maintenance activities or specifically to reduce maintenance effort. Other motivations for improvement efforts that respondents cited included reducing development costs (61%), shorten development cycle times (61%) and improve management visibility in the development process (38%).

We then asked respondents how the success of improvement programmes was judged and monitored. Table 3 summarises the success criteria reported by respondents.

Table 3. Success criteria of process improvement

Factors used as success criteria for process improvement

	Frequency	Percentage
Reduced development effort	20	24
Reduced maintenance effort	14	16
Improved defect detection	23	27
Improved operational reliability	15	18
Reduced delivered defects	25	29
Improved documentation	12	14

NB Respondents were at liberty to choose multiple criteria

Table 3 shows that, of the success criteria offered in the questionnaire, 29% of respondents selected 'reduced delivered defects' as a criteria - making it the most popular criteria, closely followed by 'improved defect detection'. Clearly both of these success criteria are highly related. However only 16% of respondents reported 'reduced maintenance effort' as a success criteria.

Further analysis of the data revealed that only 36 of the 85 companies (42%) selected any criteria at all. That means that more than half of the companies in our sample had no explicit success criteria for process improvement.

These are important findings as they suggest that many companies are not only poor at identifying problems that improvement effort will be directed towards but also companies fail to monitor the impact of improvement efforts.

3.2.2. Maintenance-oriented data collected by companies

Collecting measurement data can help companies to plan and monitor maintenance by, for example, monitoring fault delivery or predicting the impact on maintenance of delivering a system at a certain point in time [14, 8]. At a most basic level companies need to be know about product faults and change data [14]. In addition companies also need to collect data characterising maintenance activities, including effort, so that progress can be monitored.

We asked companies via our questionnaire about the measurement data they collected that related to the planning and management of maintenance.

Table 4 presents data characterising the general use of metrics data within our sample of 85 companies. It shows that, while most companies in our sample collect some measurement data, only 19% use a lot of data. Although, on the face of it, this does not particularly bode well for the control and management of maintenance activities it is probably a reasonable result in that the CMM only expects mature companies to make extensive use of metrics data.

Table 4. The extent of metrics use

The extent of metrics use

	Frequency	Percentage
Major	14	16
Minor	53	62
None	6	7
Missing	12	14
Total	85	100

In our case studies developers regularly cited effective configuration management as critical to the efficient deployment of maintenance. Many practitioners told us they believed that good configuration management processes made maintenance activities much less effort. Table 5 shows that despite this, less than half of the companies in our study appear to be monitoring their configuration management processes. Given the significance that practitioners ascribe to configuration management, this result suggests that companies are not monitoring configuration management activities optimally. Surprisingly, further analysis of this result did not reveal a clear statistical correlation between companies with high maturity levels and whether configuration management data is collected, nor between formal CMM assessment and configuration management data.

Table 5. Extent of configuration management data collected

Collection of configuration management data

	Frequency	Percentage
Collected	35	41
Not collected	7	8
Missing	43	51
Total	85	100

The collection and analysis of data characterising the faults discovered during development and testing is believed to be a reliable predictor of residual faults delivered to customers [6]. Clearly residual faults contribute significantly to maintenance effort. Such data also contributes valuable information about when a product is ready to be delivered to customers. Fault analysis data can play an important role in planning and managing maintenance activities [6]. Table 6 shows that, although, almost half of our sample do some sort of fault analysis, a significant proportion of companies do no fault analysis (assuming that missing responses indicate that no such data is collected). Again this result suggests that companies are failing to collect the data that will help them plan and manage their maintenance activity.

Table 6. Extent of fault analysis data collected

Collection of fault analysis data

	Frequency	Percentage
Collected	41	48
Not collected	1	1
Missing	43	51
Total	85	100

We also asked respondents whether their company collected data that could more indirectly aid the planning and management of maintenance activities. For example we asked whether data was collected measuring:

- Re-work effort
- Peer reviews
- System maintainability
- System complexity

Our results identify very few companies that collect and use such data.

4. Discussion of Results

Although it is widely believed that maintenance accounts for upwards of 70% of software engineering effort [4], our results show that companies are not controlling the maintenance implications of problems that are encountered during software development. Software professionals in our study were aware of the significant maintenance burden being experienced by their company and they were also aware of their company's desire to reduce maintenance effort. However awareness of both of these things seemed rather informal and it was difficult to find a company that actually had in place explicit measures to monitor maintenance activity. It seemed that although most companies paid 'lip service' to reducing maintenance and solid measures to reduce maintenance effort were not in place in many companies. Indeed we got the impression from case study participants that companies believed actually reducing the maintenance burden would involve fundamental changes to the development process that would prove too costly in the short term. The following quote from a software developer in one company sums up the frustration that many of our study participants voiced to us:

"…software is being released with known faults as time usually takes precedence over quality."

Our results also show that companies did not seem to have in place measures to identify the causes of maintenance effort. Companies did not seem to be making explicit links between problems experienced in other areas of the development process and the maintenance consequences of those problems. In our study it was difficult to find companies with a rigorous approach to controlling and managing projects with the aim of reducing maintenance effort. Even companies that acknowledged that they had problems with, for example, user requirements capture, seemed to think of this problem in isolation rather than link this explicitly with subsequent maintenance implications.

Our case study results show that companies have most problems with 'soft issues' in software development and have much fewer problems with technical issues. Organisational issues contributed many problems to the development process and even within the development process user requirements capture was most problematic. Companies reported encountering very few problems with the technical areas of system design and coding. This supports the views of Bennett and McDermitt [7] when they call for a re-assessment of the balance of software engineering research towards soft issues.

Few companies in our study monitored and controlled their maintenance activities effectively. Most did not explicitly include the reduction of maintenance effort as an improvement goal. However a few companies did. It may be that those companies are the higher maturity companies (we will need to do further analysis to determine this). However it is interesting to note that the vast majority of the 85 companies in our survey were ISO9001 certified [12].

5. Future work

We have collected a great deal of data that we have yet to analyse. We need to extend the analysis presented here to include all thirteen of the case study companies. We particularly need to analyse our data further to see whether it reveals the characteristics of the few companies that manage maintenance effectively. So far our relatively small sample has produced correlations with unacceptable levels of statistical significance. For example, our data currently does not confirm that higher maturity companies manage maintenance more effectively than lower maturity companies. This may be because our sample size is too small or it may be for reasons related to maturity assessments. It is a result we need to follow-up. Such a follow-up will determine whether our results simply reflect the fact that an estimated 70% of companies remain at CMM level one [http://www.sei.cmu.edu/cmm/].

Other interesting questions have emerged from the analysis we present here that also need following-up, for example it would be valuable to ask practitioners to actually estimate how much time they personally, and their company generally, spends on maintenance.

We also need to perform a detailed analysis of the way formal process models address maintenance before contributing to the on-going debate in this area that we hinted at earlier. In particular, the question of whether it is effective to separate development from maintenance is an area that could be usefully explored more fully in the future.

6. Conclusions

We have shown in this paper that although conventional wisdom tells us that maintenance is a very important issue in the software industry many companies do not appear to be taking steps to address this. Furthermore we have shown that there appears to be a conflict between what companies say they want to do regarding reducing maintenance effort and the actual effort they put into its reduction. It may be that companies are aware that they need to reduce maintenance effort, but are unsure about how to achieve this.

Our results also show that most of the problems that impact on software development are related to organisational issues rather than technical issues. This finding supports the view that there should be a re-balancing of software engineering research effort so that the discipline has a more holistic understanding of software engineering activities.

Our results show that companies do not seem to be addressing maintenance explicitly in their software improvement efforts. This is despite the fact that practitioners at all levels in companies reported that maintenance was a significant problem for them and their company. The objectives of many companies' improvement activities seemed vague and unfocussed, popular objectives included the ubiquitous 'improving software quality'.

Overall, our results show that although some companies seem to be in a good position to reduce their maintenance burden, most companies lack a strategic focus linking the causes to effects in maintenance. We believe that with a little more strategic direction companies could reduce their maintenance burden.

Acknowledgements

We are sincerely grateful to all the companies and practitioners (who, for reasons of confidentiality, must remain anonymous) for their participation in this project. The project is funded by the UK's Engineering and Physical Science Research Council, under grant number EPSRC GR/L91962.

References

[1] El Emam K, Laitenberger O, Harbich T (2000) "The Application of Subjective Estimates of Effectiveness to Controlling Software Inspections" Journal of Systems and Software, 54, pp119-136

[2] Hall T, Baddoo N, Wilson D (2000) "Measurement in software process improvement programmes: an empirical study" IWSM2000, Springer Verlag

[3] Hall T, Wilson D (1997) "Views of software quality: a field report" *IEE Procs on Software Engineering*, April, pp111-118

[4] Hanna M (1993) "Maintenance Burden Begging for a Remedy" Datamation April 1993, pp53-63

[5] Kuilboer JP, Ashrafi N (2000) "Software Process and Product Improvement: An Empirical Assessment" Information and Software Technology Journal 42 pp27-34

[6] Littlewood B, Strigini L (2000) "Software reliability and dependability: a roadmap" International Conference on Software Engineering, IEEE proceedings

[7] McDermid, J.A.; Bennett, K.H. (1999) "Software engineering research: a critical appraisal" IEE Proceedings- Software, 146(4) , Aug. pp179 - 186

[8] Offen JR, Jeffery R (1997) *"Establishing Software Measurement Programs"* IEEE Software, March/April, pp45-53

[9] Wilson D, Hall T, Baddoo N (2001) "A Framework for evaluation and prediction of software process improvement success" *Journal of Systems & Software* (to appear)

[10] Berdie DR and Anderson JF (1974) *Questionnaires: Design And Use*. The Scarecrow Press, Metuchen.

[11] Humphrey WS (1989) *Managing The Software Process.* Addison Wesley.

[12] ISO (1999) *Software Process Assessment - Part 5: An Assessment Model And Indicator Guidance* International Standards Organisation.

.

[13] Paulk M, Weber C, Curtis B, Chrissis M (1997) *The Capability Maturity Model: Guidelines For Improving The Software Process.* Addison-Wesley.

[14] Pulford K, Kuntzmann-Combelles A and Shirlaw S (1996) *A Quantitative Approach To Software Measurement.* Addison Wesley

Appendix One - Characteristics of Case Study Companies

Company number	**HW/SW Producer**	**UK or Multi-national?**	**Size (people)**	**SE size (people)**	**Age (yrs)**	**SW type**	**CMM Level (self-estimate)**
1	HW/SW	MN	>2000	>2000	>50	Real time embedded	1*
2	SW	UK	100-500	100-500	20-50	Business systems	1
3	HW/SW	MN	>2000	500-2000	>50	Real time embedded	1
4	HW/SW	MN	>2000	500-2000	>50	Real time embedded	1
5	SW	MN	>2000	>2000	10-20	Real time	4*
6	SW	MN	>2000	>2000	10-20	Real time	3*
7	SW	MN	>2000	>2000	20-50	Packages	1
8	SW	UK	10-100	10-100	5-10	Business systems	2
9	SW	MN	10-100	10-100	10-20	Real time embedded	3
10	SW	MN	>2000	10-100	10-20	Embedded Systems software	1
11	HW/SW	MN	500-2000	11-25	20-50	Real time embedded	2
12	HW/SW	UK	100-500	<10	20-50	Embedded	1
13	SW	UK	100	40	10-20	Business systems	3

* formal CMM assessment

Appendix Two - Demographic data from questionnaire responses

Table a1 Scope of company

Scope of company

	Frequency	Percentage
Multi-national	46	55
UK based	38	45
Total	84	100
Missing	1	

Table a2 Size of development effort

Size of software development function

Staff numbers	Frequency	Percentage
0 - 25	38	45
26-100	22	26
101+	23	27
DK	1	1
Total	84	100
Missing	1	

Table a3 Age of company

Age of company

Years	Frequency	Percentage
0-5	2	2
6-10	12	14
11-20	34	40
20+	37	43
Total	85	100
Missing	0	

Table a4 ISO certification status

ISO 9001 certified

	Frequency	Percentage
Yes	77	95
No	4	5
Total	81	100
Missing	4	

Table a5 Effort areas

Major effort area
(>20% of all software development effort)

	Frequency*	Percentage
System development	58	68
System maintenance	32	38
User support	33	39
Computer operations	5	6
Consultancy	23	27

* Note that respondents could choose more than one major effort area.

Table a6 Application areas

Type of software developed

	Frequency*	Percentage
Bespoke systems	64	75
Commercial packages	44	52
Safety critical	24	28
Data processing	45	53
Business systems	54	63
Systems software	37	44
Telecommunications	34	40

* Note that respondents could choose more than one application area.