Mohamad Kassab[1], Giuseppe Destefanis[2]

Requirements Effort Estimation: The State of the Practice

Abstract

Estimating the effort is an important task in software project management. A realistic effort estimation right from the start in a project gives the project manager confidence about any future course of action, since many of the decisions made during development depend on, or are influenced by, the initial effort estimations. Nevertheless, little contemporary data exists for documenting actual practices of software professionals for requirements effort or size estimation. We have conducted an exploratory survey concerning the state of practice of requirements engineering. In this paper we report the results from this survey that are of particular interest to the software estimation community with an emphasis on linking the reported practices to productivity of the development process and the project's duration and budget.

Keywords: effort estimation, effort, Requirements Engineering, SDLC, agile

1. Introduction

According to the Guide to the Software Engineering Body of Knowledge (SWEBOK v3.0)[3], Requirements Engineering (RE) represents the first skill (or Knowledge Area) that a professional Software Engineer must demonstrate competency in. The SWEBOK Guide asserts that software project success is highly dependent upon the quality of the RE process[4]. In 2008 Boehm, Valerdi and Honour[5] analyzed a dataset of 161 software development projects using

1 The Pennsylvania State University, Malvern, PA, U.S.A, muk36@psu.edu

2 CRIM, Computer Research Institute of Montreal, Canada, giuseppe.destefanis@crim.ca

3 *Guide to the Software Engineering Body of Knowledge*, *Version 3.0*, eds. P. Bourque, R.E. Fairley, IEEE Computer Society 2014, www.swebok.org

4 Ibidem.

5 B. Barry, R. Valerdi, E. Honour, *The ROI of Systems Engineering: Some Quantitative Results for Software-Intensive Systems*, "Systems Engineering" 2008 (Wiley Periodicals, Inc), vol. 11, no. 3, pp. 221–234.

the COCOMO II software cost estimation model and demonstrated that those projects where considerable effort was expended in up-front requirements and architectural definition realized an average of 92% in cost savings compared to those projects where minimal effort was expended on these activities.

As a practical matter, it is typically useful to have some concept of the "volume" of the requirements for a particular software product. This number is useful in evaluating the "size" of a change in requirements, in estimating the cost of a development or maintenance task, or simply for use as the denominator in other measurements[6].

Even though there is ample information available on solid RE estimation practices, the software development industry, as a whole, has a disappointing track record when it comes to completing a project on time and within budget. The Standish Group published its well-known Chaos Report[7] in 2014 in which it was noted that only 16.2% of the surveyed software projects were completed successfully within the estimated schedule and budget. In larger companies, the news is even worse: only 9% of their projects come in on-time and on-budget. And, even when these projects are completed, many are no more than a mere shadow of their original specification requirements. The results also indicate that 52.7% of projects will cost 189% of their original estimates. According to the same report; proper planning and realistic expectations are among the major success criteria for a software project (ranked fourth and fifth among top ten criteria)[8].

On the other hand, it is true that RE practices in general and RE estimation practices in particular have been challenged in the past years. A fast changing market demands software products with ever higher reliability, higher performance, and higher security among many other quality requirements. But the quality requirements are very challenging when estimating the effort and the time it would take to implement them[9]. This is mainly because of the unique nature of these requirements: Quality requirements are subjective, relative, interacting and crosscutting. However, it is still crucial to be able to make decisions about the scope of software by given resources and budget based on a proper estimation of building both Functional Requirements (FRs) and quality requirements.

---

6 A. Watt, *Project Management*, BC Open Textbooks 2014.

7 Standish Group, *The CHAOS Report 2014*, http://www.projectsmart.co.uk/docs/chaos-report.pdf

8 Ibidem.

9 L. Chung, B.A. Nixon, E. Yu, J. Mylopoulos, *Nonfunctional Requirements in Software Engineering*, Kluwer Academic Publishing 2000.

With the increasing popularity of agile approaches, the RE discipline was also challenged under the agile umbrella. This is basically true because "making requirements" requires spending more time studying the business and application world and less time programming. Making requirements takes time – the time that could be spent writing code – and users often feel better if they think that people are "working" (that is, coding) on their problems[10]. In fact, ever since agile first emerged in the 1990 s, there has been debate about what role RE can play for agile practitioners and their customers[11]. Williams[12] discusses some of the shortcomings in using agile methodologies with respect to RE. Since the requirements effort estimation activity is an integrated part of the RE discipline, it inherits the above challenges as well.

The key issue in implementing an improvement in industrial practices is to first identify the areas that need most improvement. In this paper, we present partial results from an exploratory survey we have conducted on the RE state of practices. We filter the responses with a focus on the requirements estimation dimension. Key questions that we aim at exploring in this paper are: What requirements estimation techniques are currently being practised in the industry? How are these techniques broken down by application domains? And how are these techniques broken down by software development life cycles (SDLC) (e.g. agile vs. waterfall)? Are non-functional requirements (NFRs) being considered while estimating the effort? And what is the link between the estimation techniques practices and completing the project on time and within budget?

The remainder of the paper is organized as follows: Section 2 describes the survey design along with the characteristics of the surveyed participants and projects. Section 3 presents the results on the current industrial practices with respect to software effort estimation. Section 4 presents related work; and finally, Section 5 concludes the results of the study presented in this paper and provides further research directions.

---

[10] K. Orr, *Agile Requirements: Opportunity or Oxymoron?*, "IEEE Software" 2004, vol. 21, issue 3.

[11] E. Letier, J. Araujo, R. Gacitua, P. Sawyer, *First International Workshop on Agile Requirements Engineering*, The European Conference on Object Oriented Programming (ECOOP), Lancaster 2011.

[12] L. Williams, *Agile Requirements Elicitation*, http://agile.csc.ncsu.edu/SEMaterials/AgileRE.pdf (April 2015).

2. Survey Design; Participants and Projects' Characteristics

We created a web-based surveys that were implemented using the web-based QuestionPro survey tool (www.QuestionPro.com). The survey consisted of 32 questions and focused on the RE State of Practice. A summary of the survey questions may be viewed through the link: https://goo.gl/h79NtF. While partial preliminary data from this survey were published in *State of Practice in Requirements Engineering: Contemporary Data* [13]; in this paper, we present additional pragmatic results with a focus on the effort estimation view.

Participants of the survey were drawn from multiple sources; including: 1) a database of former students in the Masters of Software Engineering degree program at the Penn State Great Valley School of Graduate Professional Studies (PSGV); PSGV caters primarily to working professionals; 2) subscribers to the IEEE Reliability Society newsletter and members of relevant Linked-In professional groups. We collected survey data through April and June 2013. Of the 373 who viewed the survey, there were 243 who started taking the survey. Of these survey takers; there were 117 who completed the survey all the way to the end. The completion rate was 48.15% and the average time taken to complete the survey was 16 minutes. The survey drew participants from 23 different countries.

The survey respondents described themselves as programmers/developers and software engineers 32.17% of the time. 44.36% of the respondents characterized themselves as architects, project managers, analysts, consultants or systems engineers; positions typically involved the higher-level aspects of the computerized system's technical design.

The survey response captured a diverse mix of project domains as Figure 1 shows. The respondents of the survey also came from a wide range of work environments in terms of the number of staff and the annual budget. The respondents also had experienced a varying number of projects in the last five years from two to more than 50. The mean experience level in this survey in terms of the number of projects in the last five years is 12.5. Nevertheless, the respondents were asked to base their project responses on only one project that they were either currently involved in or had taken part in during the past five years. Reported projects were distributed across a broad range of categories (Figure 2).

---

[13] M. Kassab, C. Neill, P. Laplante, *State of Practice in Requirements Engineering: Contemporary Data*, "ISSE" 2014, vol. 10, no. 4, pp. 235–241.

As for the projects' duration from inception to delivery, 36% of the projects took 6 months or less to complete with 23.33% taking between 6 months and 12 months, 15.33% taking between 12 months and 18 months to complete and 6% taking between 18 months and 24 months to complete. Only 19.33% of the reported projects took more than 24 months to complete.

The average number of full time staff (IT) that were involved in the project all together is 15.

For those projects that followed an agile development method (e.g. SCRUM, Extreme Programming, Feature Driven Development etc.), 26.67% of these projects took less than 5 iterations to complete, 16% took between 6 and 10 iterations to complete, 11.33% took between 11 and 15 iterations to complete, 2.67% took between 16 and 20 iterations to complete and 9.33% of these projects took more than 20 iterations to complete. The average duration in business days for one iteration in an agile project was 21.76 business days.

The survey also looked at project size in terms of lines of code. The projects experienced were predominantly of small to medium size in terms of lines of code. 54.96% of the projects contained 50,000 lines of code or less.

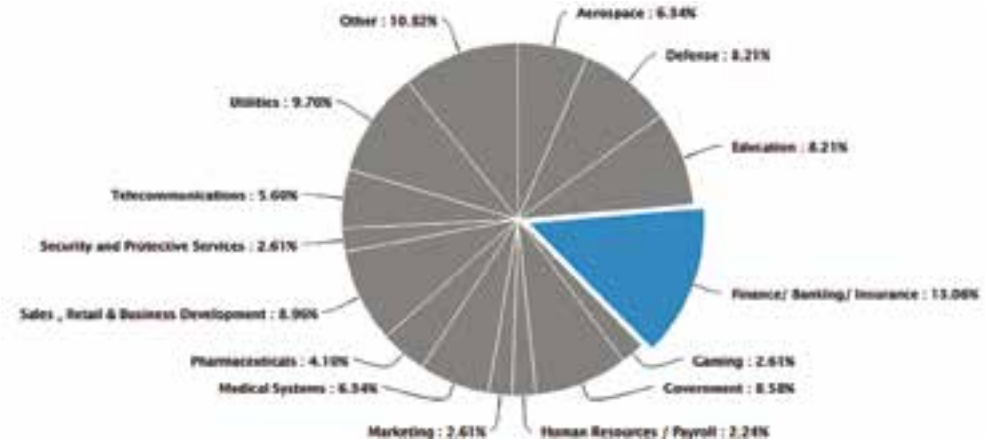

Figure 1. Projects' Domains Distribution Reported in the Survey (n = 243)

Source: the authors' own study.

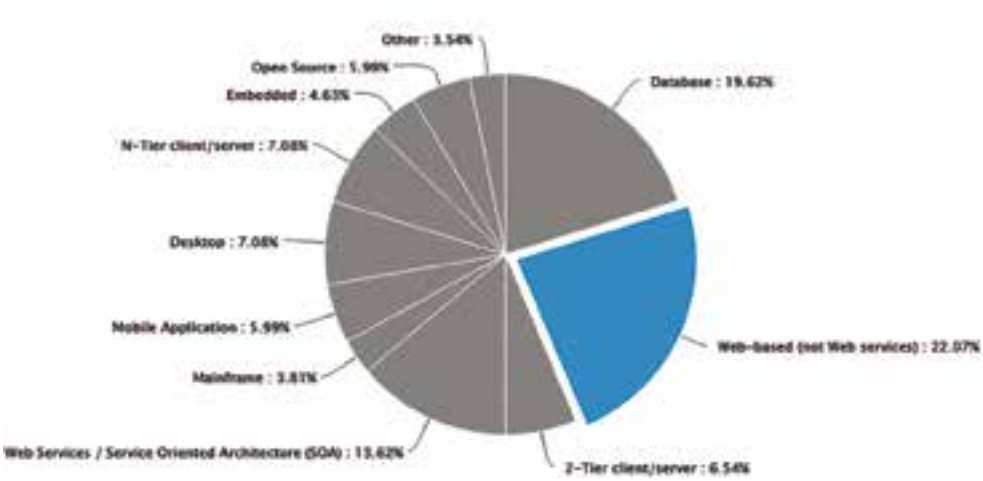

Figure 2. Projects' Categories Distribution Reported in the Survey (n = 243)
Source: the authors' own study.

3. State of the Industrial Software Effort Estimation Practice

We filtered the RE industrial practices surveyed by the questions related to the requirements effort estimation. We further linked the responses from effort estimation questions to other questions in the survey related to other RE practices and overall project productivity. In this section, we present the results and our own observations.

Sixty percent of the respondents reported on performing estimation for the size of requirements or the effort of building them. Out of this group of respondents, 62% reported on taking into account the NFRs during the size/effort estimation. When we broke these numbers down across the SDLC practices, we found that 67% of the agile projects reported on performing estimation (comparing to 45% for the waterfall). And then we found that 68% of the agile population considered NFRs during the estimation (comparing to 40% for the waterfall). This was a surprising finding, as there is a common belief that agile methodologies deal less with NFRs compared to the waterfall[14]. One reason for this belief

[14] L. Williams, *Agile Requirements Elicitation*, http://agile.csc.ncsu.edu/SEMaterials/AgileRE.pdf (April 2015).

is that NFRs are not always apparent when dealing with requirements functionality only (through user stories).

On a different note, we noticed that when the project size increased, there was more agreement that at least one size/effort estimation technique was executed on the requirements (see Figure 3). We could not find the same trend when comparing with the overall size of the organization in terms of the annual budget or number of employees.

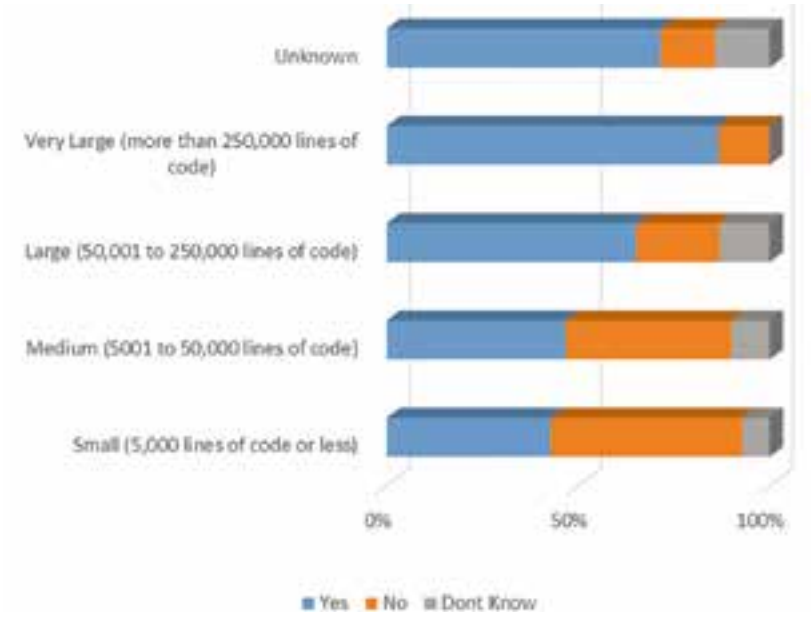

Figure 3. Link between the Question: "Did you perform an estimation for the size of requirements or efforts of building them?" and the Project Size (n = 219)

Source: the authors' own study.

For those who conducted estimation for the size/ effort, "Expert Judgments" was the most popular technique (24%) (See Figure 4).

Expert judgment was the dominant technique for the waterfall projects (see Figure 5). While it was still popular for agile projects (ranked second for these projects), we see the emerging of "Group estimation" techniques such as planning poker and wideband Delphi in the agile projects. We also see the emerging of "story points" in the agile projects (ranked third) compared to their usage in the waterfall projects (ranked 5th). Only a small proportion of this survey population reported on using any formal size/effort method (e.g. COCOMO, COSMIC, function points). That was the case for both the agile and waterfall projects.

Figure 4. Distribution of the Effort Estimation Techniques Employed across the Surveyed Projects (n = 131)

Source: the authors' own study.

We also see less usage of "use case points" in the agile projects compared to the waterfall projects (see Figure 5). This finding is expected as there is more reliance on the use-cases to represent requirements in the waterfall comparing to the agile.

The questions relating to delivery timeline, schedule and costs indicate that the projects represented in this study took longer than the respondents had expected to deliver. Only 48% of the respondents agreed that the duration of the project was within schedule; and only 21% agreed that the project goals were achieved earlier than predicted. Also only 45% agreed that the project costs were within budget estimates.

For the subgroup of respondents who took NFRs into consideration when estimating the effort; 55% reported an agreement that the duration of the project was within schedule; and 52% reported an agreement that the budget costs were within budget estimates (comparing to only 36% and 25% from the projects that did not account for NFRs). This finding is aligned well with what we proved in an earlier work that the lack of effort estimation approaches which take into account the effect of the NFRs on early effort estimation contributes to the Cone of Uncertainty phenomenon for a project; and it leads to delays and running over budget[15].

[15] M. Kassab, *Non-Functional Requirements: Modeling and Assessment*, VDM Verlag Dr. Mueller 2009.

When they were asked whether corrective hours resolving run-time problems were minimal, 49% of the respondents expressed their agreement. On the other hand, 45% of the respondents agreed that the project could have been completed faster, but that would have meant a lower quality product.

Overall, 46% of the respondents indicated that they used an agile methodology such as SDLC for their project making agile the most popular SDLC[16]. This is aligned well with the popular belief that agile development practices have become widely accepted as an effective class of approaches to project management in order to have a rapid delivery of high-quality software[17]. Usually the work of the teams adopting an agile methodology is flexible and changeable. In comparison to traditional software processes, agile development is less document-centric and more code-oriented, adaptive rather than predictive, and people-oriented rather than process-oriented[18]. In this survey, the agile projects outperformed the waterfall projects in maintaining the project within the schedule and budget (see Figure 6).

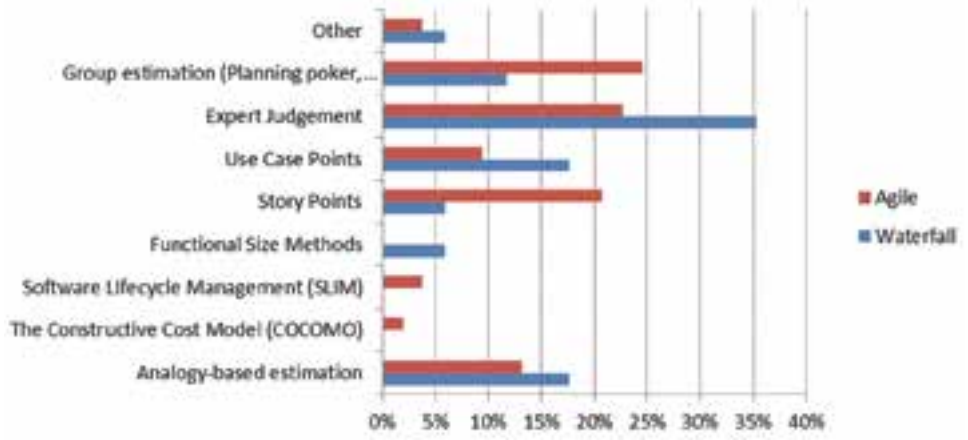

Figure 5. Effort Estimation Techniques Used across Agile vs. Waterfall (n = 85)

Source: the authors' own study.

[16] M. Kassab, C. Neill, P. Laplante, *State of Practice in Requirements Engineering: Contemporary Data*, "ISSE" 2014, vol. 10, no. 4, pp. 235–241.

[17] T. Huston, *What is Software Testing? A Brief Overview of the Agile Approach to Software Development and Testing*, http://smartbear.com/products/qa-tools/what-is-agile-testing/ (May 2015).

[18] Agile Manifesto, http://agilemanifesto.org/ (May 2015).

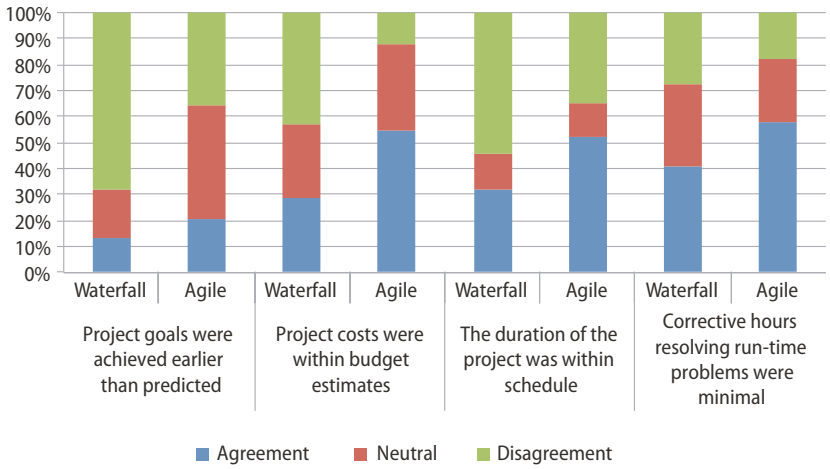

Figure 6. Reported Level of Satisfaction on the Productivity in Agile vs. Waterfall Projects (n = 82)

Source: the authors' own study.

We further linked the question of estimation techniques to whether the duration of the project was within schedule (see Figure 7). Considering the results from this linkage and since the waterfall projects reported more reliance on experts' judgment and analogy based approaches compared to the agile projects that reported more reliance on Group estimation and Story Points (see Figure 5), we believe this may be an important factor explaining our other finding on agile outperforming (compared to the waterfall) in the project's duration and budget estimation (see Figure 6).

We broke down the responses to the question "Did you perform an estimation for the size of requirements or efforts of building them?" by application domains of the projects. While most of the domains reported more than 50% agreement (see Figure 8), we notice that the gaming industry was standing last in performing any effort estimation on requirements (at only 33% agreement). It was surprising, though, to find that the level of satisfaction within the gaming applications domain was at 100% with the overall efforts to properly estimate the size / effort on requirements (see Figure 9).

By analyzing further the estimation techniques employed at the different domains, we observe that "Story Points" is a common technique for the majority of domains (13 out of 14 domains employed this technique). "Story Points" was popular in Aerospace applications in particular (at 50%). "Education projects" reported the highest level of utilizing formal estimation techniques comparing to other domains (at only 25%).

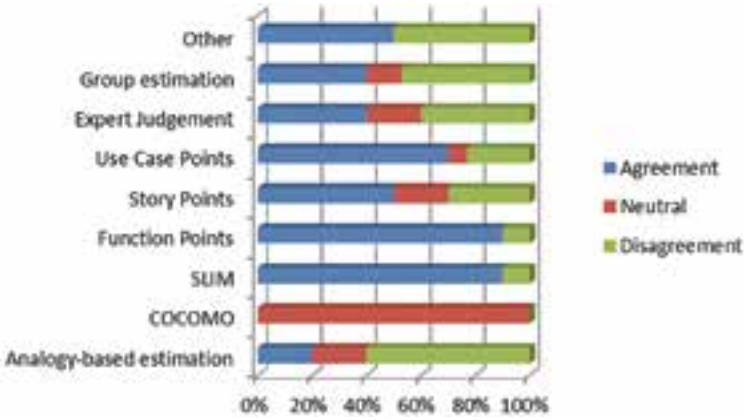

Figure 7. Link between the Effort Estimation Technique and the Statement: "The duration of the project was within schedule" (n = 84)

Source: the authors' own study.

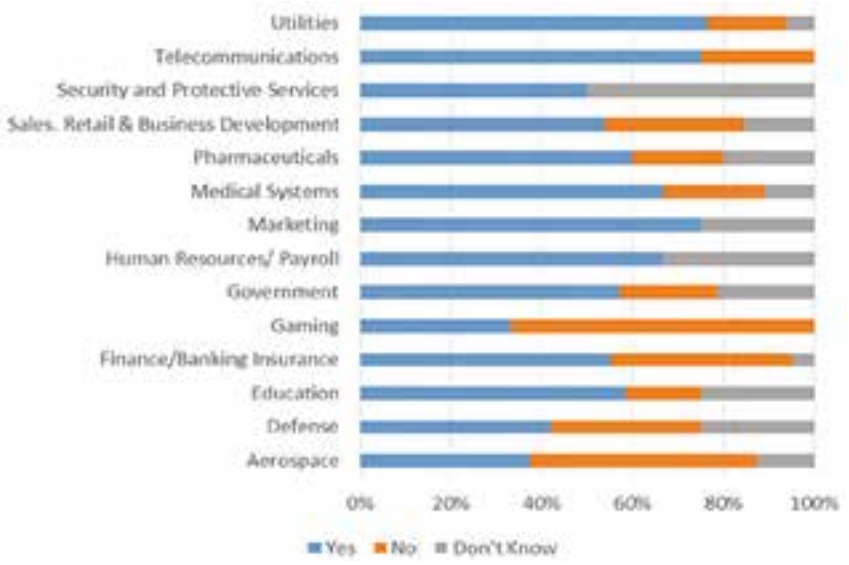

Figure 8. Link between the Question "Did you perform an estimation for the size of requirements or efforts of building them" to the Application Domain (n = 219)

Source: the authors' own study.

Remembering that the waterfall projects reported higher usages for "Expert Judgement", "Analogy Based" and "Use Case Points", we were interested to link this finding to the earlier one that the "Finance/Banking" industry was the only

domain that reported higher adoption for waterfall SDLC compared to the agile[19]. We notice indeed that these three estimation techniques dominated the overall estimation techniques for the finance/banking domain (at 65%).

As whether the duration of the project was kept within schedule, both Aerospace and the Telecommunications reported the worst performance (at only 37.5% agreement rate).

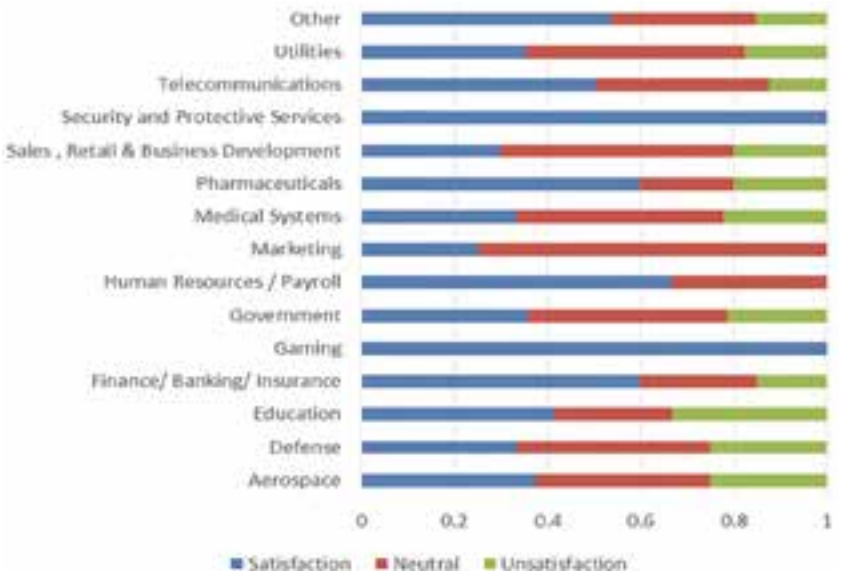

Figure 9. Link between the Satisfaction Level with Requirements Estimation Techniques Applied and the Application Domain (n = 81)

Source: the authors' own study.

4. Related Work

There are many attempts to investigate the RE state of practice in general. In 2003 Neill and Laplante presented results of a comprehensive survey showing RE practices across a broad range of industries and project types[20]. The

[19] M. Kassab, C. Neill, P. Laplante, *State of Practice in Requirements Engineering: Contemporary Data*, "ISSE" 2014, vol. 10, no. 4, pp. 235–241.

[20] C. Neill, P. Laplante, *Requirements Engineering: The State of the Practice*, "Software" 2003, vol. 20, no. 6, pp. 40–46.

survey results were highly cited (181 times via Google Scholar at this writing), and seemed to provide a template for more focused requirements surveys. For example, Khurum et al.[21] conducted a brief survey to uncover challenges in organizations to effective RE, Chernak[22] surveyed a small group of companies to determine the prevalence of requirements reuse and Verner et al.[23] sought to uncover specific issues with respect to requirements management. The literature is rich with other survey studies on RE practices, development needs, and preferred ways that target local geographic locations (e.g. Finland[24], Chile[25], and Malaysia[26]). The data from these surveys were ignorant to the requirements effort/size estimation activity in particular. To remedy this deficiency and provide useful data to other researchers we updated and reprised the Neill and Laplante 2003 survey and included questions related to effort/size estimation in the survey discussed herein.

The authors presented the results of a series of industrial surveys aimed at analyzing industrial practices with respect to modeling (estimation and measurement) software development effort and productivity[27]. In *State of the Practice in Software Effort Estimation: A Survey and Literature Review*[28], the authors

---

21 M. Khurum, N. Uppalapati, R. Chowdary, *Software Requirements Triage and Selection: State-of-the-Art and State-of-Practice*, 19th Asia-Pacific Software Engineering Conference 2012, pp.416, 421.

22 Y. Chernak, *Requirements Reuse: The State of the Practice*, 2012 IEEE International Conference on Software Science, "Technology and Engineering" 2012, pp. 46, 53.

23 J. Verner, K. Cox, S. Bleistein, N. Cerpa, *Requirements Engineering and Software Project Success: An Industrial Survey in Australia and the US*, "Australasian Journal of Information Systems" 2007, vol. 13, no. 1.

24 U. Nikula, J. Sajaniemi, H. Kälviäinen, *A State-of-the-Practice Survey on Requirements Engineering in Small- and Medium-Sized Enterprises*, Research report, Telecom Business Research Center Lappeenranta 2000.

25 A. Quispe, M. Marques, L. Silvestre, S.F. Ochoa, R. Robbes, *Requirements Engineering Practices in Very Small Software Enterprises: A Diagnostic Study*, *CCC 2010*, Proceedings of the XXIX International Conference of the Chilean Computer Science Society, Antofagasta 2010.

26 B. Solemon, S. Sahibuddin, A.A. Abd Ghani, *Requirements Engineering Problems and Practices in Software Companies: An Industrial Survey*, "Software Engineering Communications in Computer and Information Science" 2009, vol. 59, pp. 70–77.

27 A. Trendowicz, *Software Effort Estimation – Overview of Current Industrial Practices and Exiting Methods: Technical Report 06.08/E*, Fraunhofer IESE, Kaiserslautern 2008; A. Trendowicz, J. Münch, R. Jeffery, *State of the Practice in Software Effort Estimation: A Survey and Literature Review*, "Software Engineering Techniques Lecture Notes in Computer Science" 2011, vol. 4980, pp. 232–245.

28 A. Trendowicz, J. Münch, R. Jeffery, *State of the Practice in Software Effort Estimation: A Survey and Literature Review*, "Software Engineering Techniques Lecture Notes in Computer Science" 2011, vol. 4980, pp. 232–245.

presented the aggregated results of two surveys and one industry workshop they conducted. The authors had a similar finding as ours that the expert's assessment is the most followed technique in estimation. Their research focus also included an assessment to the objective of the estimation and perquisite to apply the chosen technique. The results in our paper provide a different view on the requirements effort estimation.

In *A Review of Software Surveys on Software Effort Estimation*[29], the authors summarized estimation knowledge through a review of ten surveys on software effort estimation that were conducted in the period between 1984 and 2002. The observations derived from the surveys were interpreted through the following points:

- Expert estimation is the method in most frequent use, which has not changed since then according to the data from our survey.
- There is no evidence that the use of formal estimation methods on average leads to more accurate estimates.
- Project overruns are frequent, but most projects do not suffer from major overruns. The average cost overrun reported by Standish Group's Chaos Report back then (89%) was not supported by other surveys. An average cost overrun of 30–40% seems to be the most common value reported.
- Managers state that accurate estimation is perceived as a problem.
- The reasons for overruns are complex, and not properly addressed in software estimation surveys. For example, software managers may have a tendency to over-report causes that lie outside their responsibility, e.g., customer-related causes.

5. Conclusion

Throughout this paper we have reported on the current landscape of practice in requirements effort estimation. While the survey indicates that the quality of the end products is high, there are still significant budget and schedule performance issues with less than 50% of those surveyed believing that the software projects were delivered on schedule and less than 50% believing that the project in question was delivered on budget. There are obviously many factors at

[29] K. Molokken, M. Jorgensen, *A Review of Software Surveys on Software Effort Estimation*, International Symposium on Empirical Software Engineering 2003, pp. 223–230.

work here, including the rampant optimism of estimates, the paucity of estimation techniques in general, and the external pressures placed on development project timelines and budgets, but these were some of the motives for the agile movement, and we are still seeing them within agile projects, if at lower levels, so again, there is still significant work to be done. Our hope is that researchers will use this data to corroborate their own research and to motivate follow-up research studies. Our own subsequent work will include repeating the survey study in 2018 to discover the changing landscape of practice then.

One limitation of this study is the limited availability of information sources. The relatively small sample of respondents in the industrial surveys might not be representative enough of the population of the software industry. Yet, since the results of the survey largely conform to the results of the literature review and our informal experience gained during multiple industrial collaborations, we conclude that the results presented in this paper represent current trends in software effort estimation theory and practice.

References

Agile Manifesto, http://agilemanifesto.org/ (May 2015).

Barry B., Valerdi R., Honour E., *The ROI of Systems Engineering: Some Quantitative Results for Software-Intensive Systems*, "Systems Engineering" 2008 (Wiley Periodicals), vol. 11, no. 3, pp. 221–234.

Chernak Y., *Requirements Reuse: The State of the Practice*, 2012 IEEE International Conference on Software Science, "Technology and Engineering" 2012, pp. 46, 53.

Chung L., Nixon B.A., Yu E., Mylopoulos J., *Nonfunctional Requirements in Software Engineering*, Kluwer Academic Publishing 2000.

Guide to the Software Engineering Body of Knowledge, Version 3.0, eds. P. Bourque, R.E. Fairley, IEEE Computer Society 2014, www.swebok.org

Huston T., *What Is Software Testing? A Brief Overview of the Agile Approach to Software Development and Testing*, http://smartbear.com/products/qa-tools/what-is-agile-testing/ (May 2015).

Kassab M., *Non-Functional Requirements: Modeling and Assessment*, VDM Verlag Dr. Mueller 2009.

Kassab M., Neill C., Laplante P., *State of Practice in Requirements Engineering: Contemporary Data*, "ISSE" 2014, vol. 10, no. 4, pp. 235–241.

Khurum M., Uppalapati N., Chowdary R., *Software Requirements Triage and Selection: State-of-the-Art and State-of-Practice*, 19th Asia-Pacific Software Engineering Conference 2012, pp. 416, 421.

Letier E., Araujo J., Gacitua R., Sawyer P., *First International Workshop on Agile Requirements Engineering*, The European Conference on Object Oriented Programming (ECOOP), Lancaster 2011.

Molokken K., Jorgensen M., *A Review of Software Surveys on Software Effort Estimation*, International Symposium on Empirical Software Engineering 2003, pp. 223–230.

Neill C., Laplante P., *Requirements Engineering: The State of the Practice*, "Software" 2003, vol. 20, no. 6, pp. 40–46.

Nikula U., Sajaniemi J., Kälviäinen H., *A State-of-the-Practice Survey on Requirements Engineering in Small- and Medium-Sized Enterprises*, Research report, Telecom Business Research Center Lappeenranta 2000.

Orr K., *Agile Requirements: Opportunity or Oxymoron?*, "IEEE Software" 2004, vol. 21, issue 3.

Solemon B., Sahibuddin S., Abd Ghani A.A., *Requirements Engineering Problems and Practices in Software Companies: An Industrial Survey*, "Software Engineering Communications in Computer and Information Science" 2009, vol. 59, pp. 70–77.

Standish Group, *The CHAOS Report 2014*, http://www.projectsmart.co.uk/docs/chaos-report.pdf

Trendowicz A., *Software Effort Estimation – Overview of Current Industrial Practices and Exiting Methods: Technical Report 06.08/E*, Fraunhofer IESE, Kaiserslautern 2008.

Trendowicz A., Münch J., Jeffery R., *State of the Practice in Software Effort Estimation: A Survey and Literature Review*, "Software Engineering Techniques Lecture Notes in Computer Science" 2011, vol. 4980, pp. 232–245.

Quispe A., Marques M., Silvestre L., Ochoa S.F., Robbes R., *Requirements Engineering Practices in Very Small Software Enterprises: A Diagnostic Study, CCC 2010*, Proceedings of the XXIX International Conference of the Chilean Computer Science Society, Antofagasta 2010.

Verner J., Cox K., Bleistein S., Cerpa N., *Requirements Engineering and Software Project Success: An Industrial Survey in Australia and the US*, "Australasian Journal of Information Systems" 2007, vol. 13, no. 1.

Watt A., *Project Management*, BC Open Textbooks 2014.

Williams L., *Agile Requirements Elicitation*, http://agile.csc.ncsu.edu/SEMaterials/AgileRE.pdf (April 2015).