A Comparison of Software Cost, Duration, and Quality for Waterfall vs. Iterative and Incremental Development: A Systematic Review

Susan M. Mitchell and Carolyn B. Seaman
Information Systems Department, University of Maryland, Baltimore County
smitchel@csee.umbc.edu, cseaman@umbc.edu

Abstract

The objective of this study is to present a body of evidence that will assist software project managers to make informed choices about software development approaches for their projects. In particular, two broadly defined competing approaches, the traditional "waterfall" approach and iterative and incremental development (IID), are compared with regards to development cost and duration, and resulting product quality. The method used for this comparison is a systematic literature review. The small set of studies we located did not demonstrate any identifiable cost, duration, or quality trends, although there was some evidence suggesting the superiority of IID (in particular XP). The results of this review indicate that further empirical studies, both quantitative and qualitative, on this topic need to be undertaken. In order to effectively compare study results, the research community needs to reach a consensus on a set of comparable parameters that best assess cost, duration, and quality.

1. Introduction

The purpose of this systematic review is to gather and synthesize empirical evidence to provide convincing and illuminating support for software project managers who need to make informed choices about software development approaches for their projects. We have specifically examined evidence from empirical studies that compare waterfall (traditional) and IID processes with respect to development cost and duration, and resulting product quality. IID, for the purposes of this work, refers to a set of practices for building software in which the overall lifecycle is composed of several iterations, each of which grows the resulting system incrementally. Each iteration is composed of activities such as requirements analysis, design, programming and testing [1]. We, therefore, include evolutionary, incremental, and agile processes in the IID category. In contrast, our working definition of the waterfall approach refers to processes that are sequential in nature, without planned iteration among phases.

We have intentionally kept the definitions of development cost, development duration, and resulting product quality broad for the purposes of our work. Accordingly, we define development cost as any expense incurred to develop or maintain a piece of software, usually related to the labor expended. Duration is the calendar time required to develop a software product, from development commitment to deployment. Resulting product quality is the combination of internal product quality (e.g. code structure, defects, maintainability) and external product quality (e.g. functionality, usability, reliability, documentation).

The remainder of this paper is organized as follows. Section 2 explains the methodology that we used. Section 3 provides the review results and their implications. Lastly, we present our conclusions.

2. Method

The objective of our systematic review was the summation and synthesis of existing studies concerning the application of waterfall (traditional) and IID software development processes. In particular, we wished to locate empirical studies that have compared these two process types with respect to development cost and duration, and resulting product quality. The methodology that we followed is that established by Kitchenham [2].

We conducted the systematic review between June 2007 and February 2008, inclusive. The studies search portion of the review concluded in September 2007. Therefore, any studies published after that date are not included. Our search was guided by the following research questions.

978-1-4244-4841-8/09/$25.00 ©2009 IEEE

- What is the *development cost* of software produced using waterfall or its variations versus using IID?
- What is the *development duration* for software produced using waterfall or its variations versus using IID?
- What is the *quality* of software produced using waterfall or its variations versus using IID?

Studies comparing portions of processes (e.g. a particular phase) were not considered suitable comparisons. Studies that made comparisons within like processes (e.g. two or more IID processes) were also not considered suitable.

The electronic databases that we searched were the ACM Digital Library, the ACM Guide to Computing Literature, IEEE Xplore, Computer Science Index, Computer Source, Elsevier, Inspec, ProQuest Digital Dissertations and Theses, Science Direct, Wiley InterScience, and World Scientific. The proceedings that were searched (other than those included in the databases listed above) were those for the Evaluation and Assessment in Software Engineering (EASE) (formerly Empirical Assessment in Software Engineering) conference and the International Conference on Software Engineering and Knowledge Engineering (SEKE). In addition, the Google Scholar search engine was used. Search results were kept in an electronic log and the EndNote (http:\\www.endnote.com) bibliographic software package was used to store any potentially relevant studies that were located.

3. Findings and Implications

The results of our search for individual studies are given in Table 1. Using all resources, we located 30 potentially relevant documents in total, nine pairs of which were revealed to be duplicates, leaving us with 21 unique documents. We next applied a set of study selection criteria. The major criteria were that the study is written in English, it is a primary study, it has been peer-reviewed, it reports empirical results, it compares waterfall and IID processes, and it presents results concerning development cost and/or development duration and/or resulting product quality. Application of these criteria reduced the set of potentially relevant studies by ten. Lastly, we applied study quality assessment criteria. The major criteria were that the study is clearly and adequately described, its design is appropriate to its objectives, appropriate and rigorous data collection methods were used, appropriate and rigorous data analysis was performed, and the study conclusions are supported by its design, data collection methods, and data analysis. Six studies did not pass the quality assessment stage. Of the six, two were earlier versions of other documents in our list, one's waterfall process included elements of iteration, and the remaining three did not pass the actual quality assessment criteria. Our systematic review, therefore, resulted in five relevant studies.

Table 1. Search Results

Step	# of Studies
Total Documents	30
Total Unique Documents	**21**
Excluded by Selection Criteria	(10)
Excluded by Quality Assessment	(6)
Final Total	**5**

Table 2 summarizes the five papers that constitute the final results of our systematic review. Meta-analysis of study results is not feasible, as widely different parameters were used in the measurement of the dependent variables of cost, duration, and quality. A general synopsis of results is as follows.

Development cost was investigated by four of the five studies (Studies 1, 3, 4, and 5 in Table 2). Two studies (1 and 5) made effort comparisons using person hours by activity (e.g. requirements specification, coding) and total person hours. Study 1 found no difference in total person hours between development models (waterfall, evolutionary, incremental, and XP), whereas Study 5 found that XP requires more total person hours than waterfall. Regarding individual activities, Study 1 found that XP generally spends less time in requirements specification than other models and Study 5 found that XP spends more time in testing than waterfall.

Two studies (Studies 1 and 4) made productivity comparisons as indicators of cost. Both studies demonstrated that XP produces code at a faster rate than waterfall. Study 3 investigated effort estimation

978-1-4244-4841-8/09/$25.00 ©2009 IEEE

Table 2. Summary of Systematic Review Results

Study	Objective	Dependent Variables	Summary of Results
1 (Benediktsson, Dalcher et al. 2006) Controlled experiment	To investigate the impact of software development approach on the resulting product and its attributes by comparing V-model (VM), evolutionary model (EM), incremental model (IM), and extreme programming (XP)	Cost: effort, productivity Quality: functionality, reliability, usability, efficiency, maintainability, defects	Cost Effort: XP groups spent significantly less time in requirements specification than V-model and evolutionary model groups. No differences in total person hours. Productivity: 1) XP produced significantly more Java LOC, 2) XP produced significantly more LOC in general than all other methodologies, 3) No differences between Java LOC per class between methodologies, 4) Significance not reported, but VM produced considerably fewer classes per project month (PM) than all other methodologies, 5) XP produced significantly more LOC per PM than VM, 6) XP produced significantly lower number of pages per PM than all other methodologies, XP produced significantly higher pages per PM than VM, no differences in total pages per PM between methodologies Quality: No reliable results
2 (Lindberg, Hsia et al. 1988) Controlled experiment	To determine if the maintenance costs of an incrementally delivered system are different from those of a conventional system	Quality: maintenance effort	Quality: 1) Person hours was mostly a function of the programmer's ability than a function of the design method, 2) No considerable differences in total LOC changed (including comments) between design methodologies, 3) No considerable differences in total LOC changed (excluding comments) between design methodologies, 4) Number of modules changed consistently higher for incremental model, 5) No considerable difference in change in program size between design methodologies
3 (Moløkken-Østvold and Jørgensen 2005) Interviews and analysis of project data	To determine if there are differences in the occurrence of effort and schedule overruns between projects following flexible and sequential development models	Cost: estimation accuracy Duration: estimation accuracy	Cost: 20% of sequential projects had accurate estimates versus 50% of flexible projects Duration: Results inconclusive
4 (Layman, Williams et al. 2004) Case study	To evaluate the effects of the use of the XP methodology vs. previous use of waterfall with some XP practices for the same product	Cost: productivity Quality: pre-release quality, post-release quality Other: team morale	Cost: 50% increase in code productivity using XP Quality: 65% improvement in pre-release quality and 35% improvement in post-release quality using XP Other: No inferences can be drawn
5 (Macias 2004) Controlled experiment	To compare XP and traditional software development approaches in terms of quality and size of the product and the time required to produce the product	Cost: person hours by activity Quality: external quality, internal quality Other: size (number of functional requirements, non-functional requirements, test cases, size of code, percent of comments)	Cost: Total time spent by teams using XP was higher than teams using traditional. Testing time spent by teams using XP was higher than teams using traditional. Quality: No statistically significant difference for either external or internal quality Other: No statistically significant differences

accuracy in relation to development model. Results showed that effort estimations for flexible (non-sequential) projects are more likely to be accurate than those for sequential (waterfall-like) projects.

Quality was investigated by four of the studies (Studies 1, 2, 4, and 5). Study 1 had no reliable results. Study 5 demonstrated no statistically significant difference in quality between the XP and waterfall models. Study 4 indicated higher pre-release and post-release product quality for XP as compared to waterfall. Lastly, Study 2 investigated product quality in terms of product maintenance effort. It found that, in general, there is no considerable difference in maintenance effort between products developed using a waterfall model and those developed using an incremental model. The authors also stated that the person hours required to make a product change is mostly a function of the programmer's ability rather than that of the product design method that was used.

Only one study, Study 3, explored duration in relation to development model. The particular aspect

978-1-4244-4841-8/09/$25.00 ©2009 IEEE

examined was schedule estimation accuracy. The authors stated that the results were inconclusive.

In summary, it can certainly be said that there is very little evidence supporting the superiority of either waterfall-based or IID models. Much of the work attempting to generate such evidence has been inconclusive, but there are a few results that support further investigation of the following propositions:

- XP projects require more effort than waterfall-based projects.
- XP projects require less effort in requirements but more in testing than waterfall-based projects.
- Programmers are more productive when practicing XP than when following a waterfall process.
- Effort estimation is more likely to be accurate for non-sequential processes than for sequential processes.
- XP results in higher quality products than waterfall-based processes.

4. Discussion and Conclusions

Formal software development models have existed for decades. As waterfall is the oldest, and is still widely practiced, it tends to be the standard against which other development approaches are compared. However, claims of its relative superiority or inferiority are overwhelmingly expressed as experience reports, anecdotes, and opinions. Indeed, in general, minimal empirical evidence exists to support the advantages of any one model over any other regarding cost, duration, or quality.

Our systematic review uncovered only five empirical studies that have compared waterfall models to IID models with regards to cost, duration, and/or quality. Four studies made cost comparisons, four made quality comparisons, and only one made duration comparisons. The parameters used for each type of comparison differed widely from study to study, eliminating the ability to effectively compare study results. For example, product quality parameters varied from subjective measures, such as usability and error handling, to objective measures, such as the number of code defects and defect density. In order to compile a preponderance of evidence supporting a model's cost, duration, and quality benefits, the research community needs to reach a consensus as to the parameters that best assess each of these aspects.

Of the five studies we located, three were controlled experiments (Studies 1, 2, and 5 in Table 2). Two used teams of undergraduate computer science students and one used what were referred to only as "well-educated" individual programmers. The use of students in academic research is a frequent practice and its shortcomings are commonly acknowledged [8]. The ability to generalize conclusions drawn from experiments using student programmers suffers from two principal realities. The first is that students are far inferior in skill level and experience to professional programmers. The second is that the products that students are able to create with their limited abilities and restricted time frames are significantly smaller, far less technologically complex, and of a much narrower domain than those developed in industry. Clearly, experiments with professionals provide richer and more generalizible results. However, we recognize the difficulties associated with conducting experiments involving professional developers. Therefore, student studies remain, out of necessity, a common source of evidence, and a valuable tool for working out the details of study design (including metrics definitions) before applying it in industrial settings.

Two of the studies (Studies 3 and 4) took advantage of historical project data. Study 3 used historical data to compare cost and duration estimation accuracy for waterfall-like and flexible (non-sequential) models. Study 4 used historical defect data to contrast the quality of a past project that used a waterfall model with a recent XP project. We see the use of historical data as a viable option for obtaining industrial, rather than student, data, although limitations exist. Additionally, open source projects may provide both past and current/ongoing cost, duration, and quality data, although categorizing the development methodologies used and obtaining the demographics of the developers may be tricky, if not unfeasible.

Another observation we have made is the almost complete lack of the collection of qualitative data. Study 3 was the only study that gathered qualitative data, using it to explore the reasons behind the difference in the quantitative results when comparing project costs. We believe that the use of qualitative methods would not only provide insight into quantitative results, but is a practical and necessary avenue to defining meaningful and consistent metrics for cost, duration, and quality comparisons. Additionally, qualitative investigation could provide the context and background needed to compare results from different studies even when different measures are used [9].

Although waterfall is the oldest software development model, it is interesting to note that only one of the five studies was from an appreciably long time ago (1988). The other four studies are relatively recent, ranging from 2004 to 2006. This appears to be because of researchers' desire to contrast waterfall-like models with more recent agile methodologies. Indeed, the remaining four studies use agile models either

978-1-4244-4841-8/09/$25.00 ©2009 IEEE

solely or in combination with other non-sequential models as their comparisons to waterfall. We acknowledge the contributions of these studies and support their continuance. However, we speculate that the comparison of models that do not diverge so extensively from waterfall may be more revealing as to waterfall's effectiveness. For example, using a process that delivers in increments, yet does not differ much otherwise from waterfall would likely deliver more convincing results because of the reduction in confounding factors.

Empirical research in the area of software development process comparison is minimal. Software organizations and project managers in particular must, therefore, rely on other (often less reliable) sources when addressing the complex, multifaceted task of selecting or modifying processes to meet their specific circumstances. There is a dire need for a credible body of empirical evidence backed by solid scientific methodology to support project managers in making optimal choices.

5. References

[1] C. Larman, Agile and Iterative Development: A Manager's Guide, Boston: Addison-Wesley, 2004.

[2] B. Kitchenham, Procedures for Performing Systematic Reviews, Keele University Technical Report TR/SE-0401, 2004.

[3] O. Benediktsson, D. Dalcher, and H. Thorbergsson, "Comparison of Software Development Life Cycles: A Multiproject Experiment," IEE Proceedings – Software, vol 153, 2006, pp. 87-101.

[4] M. Lindberg, P. Hsia, and R. Bond, "A Study of Requirements Change Effects on Incrementally Delivered Systems," Proc. of the Conference on Software Maintenance, 1988.

[5] K. Moløkken-Østvold and M. Jørgensen, "A Comparison of Software Project Overruns-Flexible versus Sequential Development Models," IEEE Trans. on Software Engineering, vol. 31, 2005, pp. 754-766.

[6] L. Layman, L. Williams, and L. Cunningham, "Exploring Extreme Programming in Context: An Industrial Case Study," Proc. Agile Development Conference, 2004.

[7] F. Macias, "Empirical Assessment of Extreme Programming," PhD thesis, Department of Computer Science, University of Sheffield, 2004.

[8] D. Sjøberg et al., "Conducting Realistic Experiments in Software Engineering," Proc. International Symposium on Empirical Software Engineering (ISESE'02), IEEE Press, 2002.

[9] C. Seaman, "Qualitative Methods in Empirical Studies of Software Engineering," IEEE Trans. on Software Engineering, vol. 25, 1999, pp. 557-572.

978-1-4244-4841-8/09/$25.00 ©2009 IEEE