International Journal of Social Science and Human Research

ISSN (print): 2644-0679, ISSN (online): 2644-0695

Volume 07 Issue 11 November 2024

DOI: 10.47191/ijsshr/v7-i11-18**,** Impact factor- 7.876

Page No: 8275-8299

How to Measure the Productivity of a Software Quality Engineer

Donald L. Buresh, Ph.D., Esq.
Morgan State University

ABSTRACT: This paper critically examines COBOL85 compiler software quality assurance project from the unintentional perspective of Total Quality. In 1988, when this project occurred, the author was not associated with TQM in any way or had any training in, or knowledge about, Total Quality. The project took place 36 years ago. The company Prime Computer, Inc. no longer exists, and the individuals involved have long since gone their separate ways. An unquestionable comfort can be derived from discussing a project that does not violate confidentiality in any way whatsoever. In 1988, this author was hired as a consultant to manage the software quality assurance effort. The charge was to significantly increase productivity in a relatively short period so that the COBOL85 compiler could be released to Prime's customers on time. This author was directly responsible for increasing the project's productivity by 40 percent over 14 weeks, while the association with the project lasted a little over a year. Unfortunately, MAI Basic Four attempted a hostile takeover of Prime, approximately three-fourths of the way through the project. Due to financial considerations and because Prime management decided to abandon their mini-computer business entirely, the quality assurance effort finished not with a bang but with a whimper. The author was then relieved of management responsibilities so that the company could save money. The last remaining software quality assurance consultant completed his work, and the COBOL85 compiler finally went to market. Because of the lack of managerial commitment, the compiler never met the expectations of the individuals dedicated to bringing to market a quality product. At the time, the author had no formal or informal training or knowledge of TQM whatsoever but did have an understanding of how a process works. The author was convinced that the process he was hired to oversee could experience dramatic improvements in productivity in a relatively short period, which could be construed as a matter of faith or possibly overconfidence. He achieved what he set out to do.

KEYWORDS: COBOL85, Plan-Do-Check-Act Cycle, Statistical Software Quality Assurance, Total Quality Management, WV Model

INTRODUCTION

This paper aims to critically examine a software quality assurance project from the unintentional perspective of Total Quality. In 1988, when this project occurred, this author was not associated with TQM in any way or had any training in, or knowledge about, Total Quality. The project took place 36 years ago. The company Prime Computer, Inc. (Prime) no longer exists, and the individuals involved have long since gone their separate ways. Thus, an unquestionable comfort can be derived from discussing a project that does not violate confidentiality in any way whatsoever.

In 1988, this author was hired as a consultant to manage the software quality assurance effort. The charge was to significantly increase productivity in a relatively short period so that the COBOL85 compiler could be released to Prime's customers on time. This author was directly responsible for increasing the project's productivity by 40 percent over 14 weeks, while the association with the project lasted a little over a year.

Unfortunately, MAI Basic Four attempted a hostile takeover of Prime, approximately three-fourths of the way through the project. Due to financial considerations and because Prime management decided to abandon their mini-computer business entirely, the quality assurance effort finished not with a bang but with a whimper.[1] The author was then relieved of management responsibilities so that the company could save money. The last remaining software quality assurance consultant completed his work, and the COBOL85 compiler finally went to market. Because of the lack of managerial commitment, the compiler never met the expectations of the individuals dedicated to bringing to market a quality product.

[1] T. S. Elliot, The Hollow Men, *Owl Eyes* (1925), *available at* https://www.owleyes.org/text/the-hollow-men/read/text-of-the-poem#root-42.

How to Measure the Productivity of a Software Quality Engineer

This paper analyzes the software quality assurance effort of the COBOL85 compiler by employing the Shiba's WV Model (WV Model)of systematic improvement, a variant of Kawakita's W Model (W Model).[2] This case study will attempt to demonstrate that the WV Model is, in some sense, canonical and that knowledge of its workings is intrinsic to process improvement. Although TQM training is essential and should never be discounted, the fact that this author independently discovered the principles of Total Quality lends credence to the notion that quality is inherent in the process under consideration and, in general, any process being conducted.

At the time, this author had no formal or informal training or knowledge of TQM whatsoever but did have an understanding of how a process works. This author was convinced that the process he was hired to oversee could experience dramatic improvements in productivity in a relatively short period, which could be construed as a matter of faith or possibly overconfidence. Even so, with these introductory remarks, this case study will begin.

TOTAL QUALITY MANAGEMENT

This section highlights total quality management. It also discusses the Total Quality methodology and describes its evolution. The following section discusses the four common threads that weave through any successful Total Quality implementation, including focusing on customers, seeking continuous improvement, requesting total staff participation, and participating in societal learning. The third section reveals that Total Quality can be viewed from an individual, workgroup, organization, or region of industry perspective. Finally, the fourth section describes the WV Model and its relationship to the Plan-Do-Check-Act (PDCA) cycle.

An Evolving Methodology

The study of Total Quality began in the 1920s and 1930s at Western Electric in Cicero, Illinois.[3] The Bell Telephone Company needed to produce reliable telephones for the burgeoning communications network they created. Walter Shewhart was mainly responsible for implementing statistical quality control procedures at the firm.[4] W. Edwards Deming, one of Shewhart's co-workers and the father of Total Quality, traveled to Japan to teach the Japanese the principles he had learned at Western Electric.[5] In a country devastated by war, the Japanese readily adopted Deming's methods and produced quality products on time.[6] The rest is history, and Japanese products are world-renowned for their quality.

According to Shiba et al., the four levels of quality are *fitness to standard*, *fitness to use*, *fitness to cost*, and *fitness to latent requirements*.[7] Fitness to standard means that a product is built according to specifications.[8] The weaknesses of fitness to standard are that it assumes that quality can only be achieved through inspection, and neglects the marketplace's needs. Both of these weaknesses exist in product development in the high-technology arena.[9] Fitness to use is the means to ensure that a product satisfies the market's needs.[10] Like fitness to standard, the weakness of fitness to use is that it is achieved through inspection. The second weakness of fitness is that a company's competitive advantage is only temporary.[11] Fitness to cost means high quality and low cost and is the essential characteristic of total quality.[12] In order to achieve this level of quality, a company must examine its production system and look for ways to improve its processes. These changes can be continuous improvement or fundamental reinvention.[13] The only weakness of fitness to cost is that competitors can create similar kinds of products that are also reliable, functional, and inexpensive to produce.[14] Fitness to latent requirements is a way of meeting a customer's needs before they know they exist.[15] Two famous examples are the Polaroid Land camera and the Sony Walkman, both of which satisfied the latent needs of their respective customers. A weakness for companies that meet fitness to latent requirements is that they may not be able to adjust quickly enough to market needs.[16]

---

[2] SHOJI SHIBA, ALAN GRAHAM, A. & DAVID WALDEN, A NEW AMERICAN TQM: FOUR PRACTICAL REVOLUTIONS IN MANAGEMENT (Productivity Press 1993) at 48.
[3] ROGER G. SCHROEDER, *OPERATIONS MANAGEMENT: DECISION MAKING IN THE OPERATIONAL FUNCTION* (McGraw-Hill, Inc.1993) at 72.
[4] *Id.* at 120.
[5] MARY WALTON, THE DEMING MANAGEMENT METHOD (Perigree Books, 1986) at 10-13.
[6] W. EDWARDS DEMING, OUT OF THE CRISIS (Massachusetts Institute of Technology, Center for Advanced Educational Services 1982).
[7] Shoji Shiba et al., *supra*, note 2 at 3-12.
[8] *Id.* at 4-5.
[9] *Id.*
[10] *Id.* at 5-7.
[11] *Id.*
[12] *Id.* at 8.
[13] *Id.* at 9-10.
[14] *Id.*
[15] *Id.* at 11.
[16] *Id.* at 12.

How to Measure the Productivity of a Software Quality Engineer

Shiba et al. recognized that Total Quality is a dynamic rather than a static concept that will continue to evolve as businesses and products change.[17] Since companies make decisions based on their corporate culture and how they perceive themselves, the notion of *fitness to corporate culture* seems to encapsulate this idea.[18] Furthermore, stakeholders are increasingly pressing firms to improve their work environment and the fitness of their products and manufacturing processes. Shiba et al. called this phenomenon *fitness for the societal and global environment*, where fitness to latent requirements is expanded to include the environment in which the customer lives.[19]

Management Thinking

Although companies implement Total Quality management differently, four common threads weave through any successful implementation.[20] They include:

- Focusing on customers;
- Seeking continuous improvement;
- Requesting total staff participation; and
- Participating in societal learning.

Companies dedicated to Total Quality focus on their customers to satisfy their needs.[21] In other words, they can react quickly, directing their limited resources to changing customer needs. Another characteristic of an organization devoted to Total Quality is that it continuously seeks to improve the quality of its products and services.[22] Continuous improvement is essentially a restatement of the scientific method, in which facts are analyzed, actions are based on facts, and results are tested empirically.[23] Furthermore, firms engaged in total quality requests that every employee, both managers and individual contributors, participate in continuously improving the company's products and services to optimize customer satisfaction.[24] If all of an organization's capabilities are involved, there is a greater probability of success. Finally, organizations involved in Total Quality need to collaborate with competitors and others to avoid reinventing methods, implement quality practices quickly, and create a quality culture to ensure that it is the norm rather than merely a passing fancy.[25]

Underlying Values of Practice

Implicit in the four different types of management thinking is the corresponding four ways to practice Total Quality. They include the:[26]

- Individual;
- Workgroup;
- Organization; and
- Region or industry.

Total Quality management must be practiced at the individual level to transform workers' attitudes from just doing their jobs to satisfying their customers by providing them with the tools to do so.[27] The idea is to bring the notion of customer/supplier relationship to every individual in a firm so that there is a shift from just carrying out daily tasks to performing daily work and effecting continuous improvement.[28] To ensure that the shift takes place, systematic labor is involved.

At the workgroup level, the daily work and the continuous improvement work must be unified so that the focus is on the process.[29] This encourages mutual learning and teamwork by creating an environment where one cannot occur without the other, and both are integral to the job. According to Shiba et al., innovative improvements should be integrated with corporate goals at the organizational level.[30] By practicing Total Quality company-wide, all of the firm's resources are mobilized to pursue quality systematically. At an industry-wide, regional, or national level, the practice of total quality is focused on ensuring that the relevant

[17] *Id.* at 26.
[18] *Id.*
[19] *Id.* at 26-27.
[20] *Id.* at 28.
[21] *Id.*
[22] *Id.*
[23] *Id.*
[24] *Id.* at 29.
[25] *Id.*
[26] *Id.*
[27] *Id.*
[28] *Id.* at 30.
[29] *Id.*
[30] *Id.*

culture supports the total quality efforts of the firm.[31] This support can come from informal networking, collaboration for mutual gain, and the transfer of successful quality practices. In Japan, the Deming Award encourages nationwide awareness about quality,[32] while the Baldrige Award performs the same function in the United States.[33] The idea is to promote the diffusion of total quality into the economic ecosphere so that customers from all walks of life benefit from the experience.

The WV Model

By focusing on the process that produces the desired results, one can decide why a process produces the actual results and how this information can be used to improve the process. This is called *management by process* and is defined below.[34]

- Set new or revised goals;
- Develop an implementation plan for accomplishing the goals
- Develop a plan for measuring whether the implementation plan is followed;
- Execute the implementation and measurement plan;
- Monitor the results and adherence to the implementation plan;
- Analyze the reasons for poor adherence to the implementation plan or poor results; and
- If necessary, begin the process all over again.

This is how management by process works. Set goals and then develop a plan to implement these goals. Create a method for measuring whether the plan is followed. Execute the plan and the measurement tool. Check the results and ensure adherence to the plan. Lastly, analyze the implementation and measurement plans to see if they worked. The implicit idea behind management by process is that "any activity can be improved if you systematically plan the improvement, understand the current practice, plan solutions and implement them, analyze the result and its causes, and cycle around again."[35]

The idea behind continuous improvement is that improvement is a problem-solving process. This process is divided into *systematic* or scientifically based improvement and *iterative* improvement.[36] Systematic improvement is based on the use of the scientific method, while iterative improvement is concerned with cycling back to work on the following problem or with continuing to improve an already improved process.[37]

Shiba et al. (1993) observed that continuous improvement differs from incremental improvement, a methodology that assumes a process is near-optimal performance and only needs minor adjustments to achieve the efficient frontier.[38] In contrast, continuous improvement is concerned with why a process behaves sub-optimally and not how far away it is from its production possibility frontier. Thus, the definition of continuous improvement is:[39]

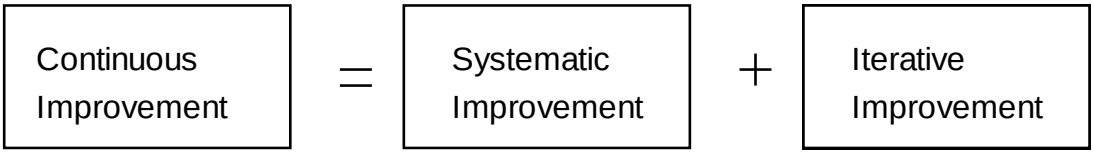

Figure 1. Definition of Continuous Improvement

The WV method of systematic improvement modifies Kawakita's W model and describes the problem-solving process as alternating from thought (rumination, planning, and analyzing) and experience (collecting data, interviews, experiments, and measurements). The name of the WV Model comes from traversing between these two levels over time and is pictured below:[40]

[31] *Id.*
[32] JAMES R. EVANS, & WILLIAM M. LINDSAY, THE MANAGEMENT AND CONTROL OF QUALITY, (South-Western College Publishing 4th ed. 1999).
[33] *Id.*
[34] Shoji Shiba et al., *supra*, note 2 at 45-47.
[35] *Id.*
[36] *Id.* at 47-48.
[37] *Id.*
[38] *Id.*
[39] *Id.*
[40] Id. at 49-50.

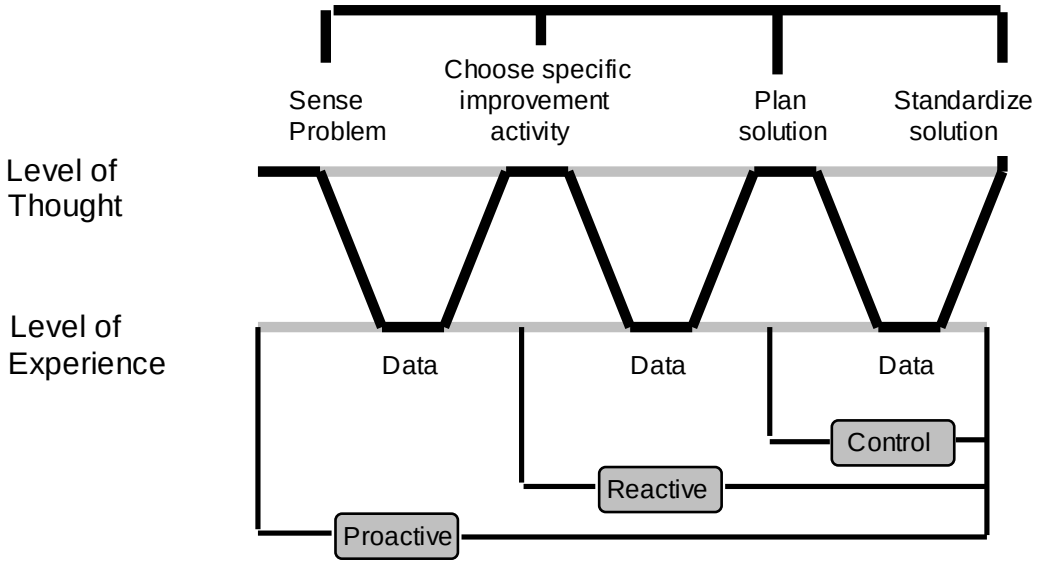

Figure 2. Problem-Solving Using the WV Model

The WV Model can also illustrate the different types of improvement: *process control*, *reactive improvement*, and *proactive improvement*.[41] Process control means monitoring a process to ensure it works correctly and adequately aligning it if necessary.[42] Reactive improvement deals with a process that is not good enough or weak.[43] Proactive improvement is about sensing a problem, exploring the situation, formulating the problem, and then improving the process.[44]

The WV Model travels from the level of thought to the level of experience by:

- Sensing a problem;
- Choosing a specific improvement activity;
- Planning a solution; and
- Standardizing the solution.

The WV Model is also known as the *Plan-Do-Study-Act (PDSA) cycle*. It is an iterative problem-solving principle where improvements are made step-by-step and where the cycle is repeated many times:[45]

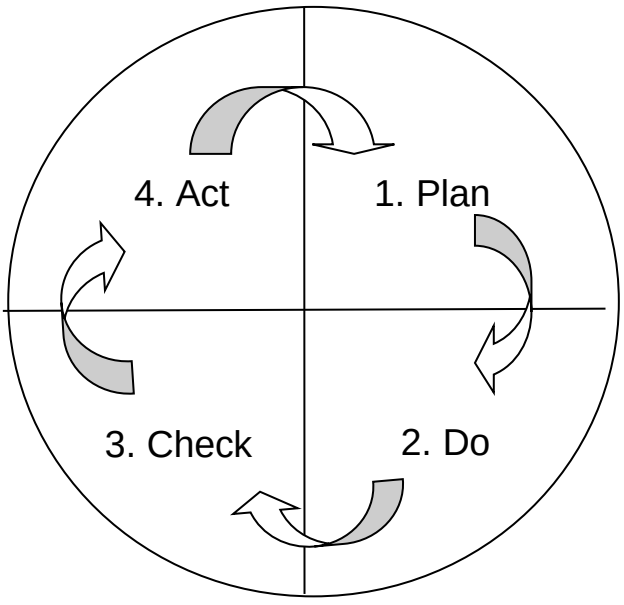

Figure 3. Plan-Do-Check-Act Cycle

[41] *Id.* At 49-54.
[42] *Id.*
[43] *Id.*
[44] *Id.*
[45] Mary Walton, *supra*, note 5 at 86.

In *planning*, the questions include what must be accomplished, the desirable changes, what new observations are needed, and how the observations will be used. In *doing* so, there is the search for data to answer the questions in the previous step or collect data on a small scale to determine the effect of the change. In *checking*, the purpose is to observe the effect of the change, and the last step is to study the results and learn from what was done. The key to *acting* is to transform how people think about their work to help satisfy the customer's needs, whether inside or outside the organization.

While managing the creation of the test bed for the COBOL85 compiler, this author, by accident, followed the PDSA cycle to improve the production of test cases. The key to the discovery was understanding the value of employing the scientific method. This came from years of mathematics, philosophy, and economics training and working part-time in a factory as an undergraduate and graduate student to pay tuition and have some money to spend. Training in this environment consisted of watching others work and doing the same.

COBOL85 QUALITY ASSURANCE EFFORT

This section discusses the COBOL85 quality assurance effort at Prime. The first subsection highlights how the author discovered the problem. The second subsection talks about selecting specific improvements. The third subsection addresses how the author planned a solution. The fourth subsection deals with standardizing the solution. The final subsection shows how the solution selected solved the software quality assurance productivity problem.

Sensing the Problem

In 1988, this author managed the computer consultants that constructed COBOL85 test cases used in quality assuring the COBOL85 compiler for Prime and was hired as a consultant a quarter of the way into the project with the sole purpose of increasing the production of test cases so that the compiler could be released in a timely manner. During this time at Prime, the current test case production effort was analyzed, new production processes were implemented and improved upon, and a statistical scheduling methodology to measure the results of our work was employed.

The first task was to become familiar with the ANSI Committee's manual for COBOL85. Prime management was interviewed to find out how they felt about the problem. They explained that the COBOL85 test cases took too long to generate. The original Test Plan projected that each test case would take, on average, 2.5 person-hours to generate, while the actual time was somewhere around 8.5 person-hours. This discrepancy was causing a severe delay in bringing the COBOL85 compiler to market, and it was the essence of this author's problem.

The next step was to interview the other consultants regarding the project. Their first reaction was that Prime management was unreasonable and demanding too much from them. The consultants said the project was delayed because Prime management wanted too many changes in the mini-specs and the test programs. They stated that Prime was not following the tenets contained in the approved Test Plan. After interviewing each consultant and reading the original Test Plan, it was discovered that it committed a grave error of omission since it called for only the following steps to take place:

- Analysis, specification, and tracking of a test case by QA
- Review of the analysis and specification by Prime technical management
- Approval of the analysis and specification by Prime technical management
- Coding and debugging of a test case by QA
- Approval of the test case by Prime technical management
- Integration of a test case by QA into Prime’s pre-existing COBOL85 test bed

These steps were missing the iterative opportunities for Prime management to adjust the analysis, specification, and source code for an individual test case.

Choosing Specific Improvements

In understanding the quality assurance effort of the COBOL85 compiler, it became apparent that the activities needed to be correctly specified to maximize the effectiveness that this author was managing. Although it goes without saying, a client is always sensitive to the amount of money expended on a contract. The technically oriented Prime employees had control over the number of test cases needed to test the COBOL85 compiler since they were the individuals who approved the test case specifications. The dollars per hour were fixed by the contracts signed by Prime and the contracting agency representing the consultants. The only variable that could be controlled was the number of person-hours worked to generate an approved test case. Therefore, the equation in Figure 4 was particularly relevant.

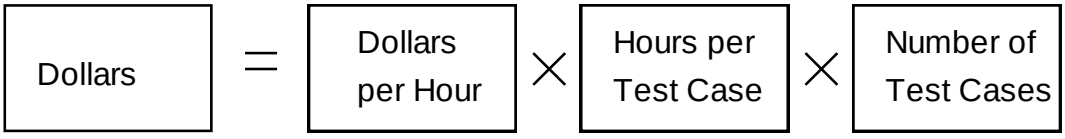

Figure 4. Breakdown of Project Dollars into Components

Since the hours per approved test case were currently at 8.5 person-hours, it was important to reduce this number significantly in a relatively short period of time.

It was imperative to model correctly the process of specifying a test case, generating the code, getting it approved by the technical people at Prime, and then putting the test case into the test bed. The three sub-cycles and the steps contained in each sub-cycle were:

- ***First Sub-Cycle***
  - Analysis, specification, and tracking of a test case by QA
  - Review of the analysis and specification by Prime technical management
  - Revision of the analysis and specification by QA; and
  - Approval of the analysis and specification by Prime technical management

- ***Second Sub-Cycle***
  - Coding and debugging of a test case by QA
  - Review of a test case by QA
  - Review of a test case by Prime technical management
  - Revision of a test case by QA
  - Approval of the test case by Prime technical management

- ***Third Sub-Cycle***
  - Integration of a test case by QA into Prime's pre-existing COBOL85 test bed

The experience taught me that the first sub-cycle, which dealt with the analysis and specification of a test case, typically occurred three times. The second sub-cycle, which focused on generating a test case, usually experienced two iterations. The third sub-cycle is not a sub-cycle but a one-time event since it integrates a test case into the COBOL85 test bed. After correctly modeling the process, it was estimated that it took approximately 5.0 person-hours on average for a test case to travel from inception to the test bed.

Planning a Solution

The next step was to examine how the consultants generated a test case to achieve the expected number of person-hours per approved test case. It was discovered that the consultants generally were not using previously approved test cases to generate new test cases. Furthermore, the naming conventions for storing test cases were not based on the associated mini-specifications written and approved during the analysis sub-cycle. The consultants resisted any changes in their work habits, feeling that they were already working at their maximum and that the real problem was with Prime management. This author implemented the necessary changes using long-suffering diplomacy, decreasing the number of person-hours per approved test case.

Another problem encountered was developing milestone schedules for Prime management. This author wanted to use a mild form of statistics to generate schedules so that Prime management could coordinate compiler development with test case production. Many hours were spent convincing both Prime management and the consultants actually writing the test cases that statistics had a place in quality assurance. After several seminal memos, Prime management approved the proposed methodology in writing and was grudgingly accepted by the individual consultants.

In the preceding paragraphs, the measure of the *average number of person-hours per approved test case* was mentioned. This measure seemed so simple at the time. Similar test cases were defined in mini-specifications. There were six or seven mini-specifications to a milestone, and this author and the other consultants were paid at the end of each milestone. Since the consultants refused to write down the number of hours they worked on any test case, measuring the actual number of person-hours expended was difficult. However, the number of test cases and billed hours were known for each milestone. Using this data, it was straightforward to calculate the average number of person-hours per approved test case.

Assuming a normal probability distribution, the only thing needed was to calculate a standard deviation. Because the consultants refused to record the time worked on a particular test case, this author used 20 percent of the mean as an operational value for the standard deviation. Now, all that was needed was to estimate the number of test cases per mini-specification and then obtain an agreement with Prime management regarding the number of mini-specifications per milestone. After Prime management signed off on these two issues, five person-hours per approved test case were employed to derive test case production schedules. The standard deviation was used to calculate confidence intervals around the estimated schedules. At the end of each milestone, a report to Prime management was produced, describing the successes and failures encountered in a milestone. What was surprising was that the workgroup of consultants met the scheduling objectives of every milestone with time to spare.

Standardizing the Solution

There was only one issue that remained. Could the production standard of 5.0 person-hours per approved test case be maintained over the long haul? At first, it was attempted to lower the number of person-hours per approved test case, but this resulted in temporarily burning out consultants, and thus, this effort was abandoned. Even so, after six months and four milestones, the average number of person-hours per approved test case stabilized at 5.0 person-hours, plus or minus 0.15 person-hours. This author was delighted with this result because it demonstrated that one could accurately schedule and measure a significant project using statistics.

Fitness Addendum

Since the COBOL85 test cases were the output of the production process, some statements must be made regarding their fitness. Because the ANSI Committee's COBOL85 Standard was employed, and with the creation of the various mini-specifications, it is evident that the test cases fit the standard. Second, since the purpose of the test cases was to exercise the functionality of Prime's COBOL85 compiler, it is also clear that the test cases were fit to use. With the increase of productivity to 5.0 person-hours per approved test case, and together with the formula expressed in Figure 4, it is evident that the test cases fit the cost. Finally, concluding that the test cases fit the latent requirements is a stretch since they were explicitly stated in the ANSI Committee's COBOL85 Standard. However, at the time, mini-computers possessed hardware and operating system (i.e., Primos) specific features, so the latent requirements for the compiler were that it be consistent with its computing environment. Both Prime management and the consulting team made every effort to ensure that these latent requirements were explicitly addressed.

MILESTONE DOCUMENTS

This section outlines the contents of Appendices A, B, and C. Each appendix is discussed in turn/

Appendix A Documents Description

At the end of each milestone, a report to Prime management was generated, indicating how many test cases were included and how many person-hours were expended. The document listed any discrepancies between what was projected at the beginning of the milestone and the actual figures. It also listed any additional test cases included in the milestone, any test case that turned out to be inappropriate, any hours employed to investigate an upcoming milestone, and any hours used for meetings. Appendix A contains a sample milestone report.

Appendix B Documents Description

Appendix B documents the revised projections for upcoming milestones. The report details the number of test cases and person-hours in each mini-specification. This report also analyzed how the person-hours were consumed to derive the number of person-hours per approved test case for the given milestone. Any estimates of future milestones and base levels were listed in the next section of the report. The estimates included each activity's expected number of person-hours and one standard deviation from the mean. On occasion, the only data available were the development times for specific activities. In this case, 0.5 and 0.8 factors were used to provide additional estimates of upcoming activities.

Appendix C Documents Description

In Appendix C, productivity estimates were calculated for the milestone that occurred prior to the arrival of this author at Prime. Specific issues that affected the measurement of productivity during this period were discussed in this document. For example, a project plan was the deliverable for Milestone I. The productivity for Milestone II was 9.4 person-hours per approved test case, while the productivity for Milestone III was 7.6 person-hours per approved test case. Since this author was hired at the end of Milestone IV, and the productivity for Milestones IV and V was approximately 7.6 person-hours per approved test case, the focus of this author's efforts was on Milestones VI and VII. The productivity for these milestones was 7.25 and 5.0 person-hours per approved test case. At the end of Milestone VII, productivity had increased by 40 percent. The productivity was then stabilized at 5.0 person-hours per approved test case so the project team could work effectively. In other words, a constant productivity measure could be used to estimate the scheduling of future activities accurately.

CONCLUSIONS

Reflecting on the project, it is essential to note that the author, despite having no prior experience in Total Quality Management, took on the responsibility of managing the test case generation for Prime's COBOL85 compiler. Formally trained as a mathematician, philosopher, and economist, the author brought a unique perspective to the project. Having participated in six quality assurance projects for different companies and products, the author understood the value of statistics and the scientific method. This commitment to employing these tools in the project demonstrates the author's dedication and expertise.

When three-quarters into the project, MAI Basic Four attempted a hostile takeover of Prime. Due to financial reasons, Prime management abandoned their mini-computer business, and the quality assurance effort seemed to slow down. After this, the author was relieved of management responsibilities so the company could save money. The last remaining software quality assurance consultant completed the test case generation, and the COBOL85 compiler limped to market. Because of the lack of

managerial commitment, the compiler never met the expectations of the individuals dedicated to bringing it to market and was deemed a failure. This was unfortunate because, from a technical perspective, the COBOL85 compiler was a world-class product, more than worthy to take its place in the high-technology arena.

DONALD L. BURESH BIOGRAPHY

Donald L. Buresh earned his Ph.D. in engineering and technology management from Northcentral University. His dissertation assessed customer satisfaction for both agile-driven and plan-driven software development projects. Dr. Buresh earned a J.D. from The John Marshall Law School in Chicago, Illinois, focusing on cyber law and intellectual property. He also earned an LL.M in intellectual property from the University of Illinois Chicago Law School (formerly, The John Marshall Law School) and an LL.M. in cybersecurity and privacy from Albany Law School, graduating summa cum laude. Dr. Buresh received an M.P.S. in cybersecurity policy and an M.S. in cybersecurity, concentrating in cyber intelligence, both from Utica College. He has an M.B.A. from the University of Massachusetts Lowell, focusing on operations management, an M.A. in economics from Boston College, and a B.S. from the University of Illinois-Chicago, majoring in mathematics and philosophy. Dr. Buresh is a member of Delta Mu Delta, Sigma Iota Epsilon, Epsilon Pi Tau, Phi Delta Phi, Phi Alpha Delta, and Phi Theta Kappa. He has over 25 years of paid professional experience in Information Technology and has taught economics, project management, negotiation, managerial ethics, and cybersecurity at several universities. Dr. Buresh is an avid Chicago White Sox fan and keeps active by fencing épée and foil at a local fencing club. Dr. Buresh is a member of the Florida Bar.

LIST OF ABBREVIATIONS

Abbreviation	Description
PDCA	Plan-Do-Check-Act
Prime	Prime Computer, Inc.
W Model	Kawakita's W Model
WV Model	Shiba's WV Model

MISCELLANEOUS CONSIDERATIONS

Author Contributions: The author has read and agreed to the published version of the manuscript.
Funding: This research received no external funding.
Institutional Review Board Statement: Not applicable.
Informed Consent Statement: Not applicable.
Conflicts of Interest: The author declares no conflict of interest.
Acknowledgments: Many thanks to Leizza Buresh for all her efforts in editing this paper. The author also thanks Dr. M. Riaz Kahn who suggested to me many years ago that I publish this paper. Any remaining issues are mine.

APPENDIX A: TYPICAL MILESTONE REPORT

To: [redacted]

From: Donald Buresh
[redacted]

CC: [redacted]

Date: June 3, 1988

Subject: Milestone Report: From 24-APR-88 to 03-JUN-88

Attached is our Milestone-8 report. During this period, 4 mini-specs (describing 125 test cases) were approved. In our analysis of Milestone-7 we projected 95 test cases for the milestone. The percent ratio of actual to expected approved test cases is 132%.

We spent 12 person-hours attending meetings. We invested 8 hours on this report that we are submitting to you. We also expended 40 person hours estimating the amount of work needed to complete BL2, BL6, BL7 and BL8. For informational purposes, we estimated the amount of computer down-time. We calculated that 36 person-hours were affected.

From April 24, 1988 to June 3, 1988 we worked 715.0 hours. We projected 480 person hours for this milestone. We calculated that we expended 17% of the contract's hours in this 6-week period. Although we made every effort to ensure that this milestone occured as close as to our predicted value in our memo dated May 2, 1988, there are four (4) reasons for the discrepancy:

1. Thirty (30) additional test cases were required.
2. Three (3) extra test cases were generated, but turned out to be inappropriate.
3. We expended 24 person hours in investigating Milestone-9.
4. Approximately 20 person hours were invested in reviewing the BL7 Functional Specification.

Had we predicted these four events during Milestone-8, we would have projected 684 person hours instead of 480 person hours. Since we expended 715.0 in Milestone-8, you can see that our productivity estimates in our May 2, 1988 memo remain accurate.

Because of differences in our billing rates, we calculated that this produces a 17.4 person hour discrepency. This figure must be subtracted from the total number of hours in the contract to determine the actual number of person hours remaining in the contract. We found that there are 1,480.6 (35%) person hours left in the contract.

Please accept the following milestone report.

Page 1 of 2

M I L E S T O N E R E P O R T

These are the activities performed by Martin Brookes, Donald Buresh, and Alex Stein from 24-APR-88 to 03-JUN-88.

I. APPROVED MINI-SPECS & TEST CASES

The mini-specifications and test cases that were approved for this milestone are:

Approved Test Cases	Expected Test Cases (1)	Actual Test Cases (2)	Percent Ratio 100*[(2)/(1)]
SC44	30	46	153%
Indirect Errors	50	61	122%
Open Output	8	12	150%
Miscellaneous	+ 7	+ 6	86%
	95	125	132%

II. MEETINGS

The meetings we attended in this milestone period are:

Meetings	Person-Hours
Department meetings:	2.0
Project meetings	10.0
Project review meetings:	+ 0.0
	12.0

III. MANAGEMENT ISSUES

- Completed this monthly milestone report for March, 1988. We expended 8 hours in producing this report.
- We continued to implement our decision to review test cases internally prior to sending them to Wendy Blatt for approval.
- For various reasons, 715.0 person hours does not accurately measure the time expended exclusively on Milestone-8 test cases. The adjusted number is 671.0 person hours. This translates into 5.2 person hours per approved test case.
- We respecified Milestone-8 and Milestone-9 for BL2. We also projected the amount of time required to generate test cases for BL6, BL7 and BL8. We expended 40 person hours to obtain this information.

IV. COMPUTER DOWNTIME (System END)

For this month, the reasons that the system was down are:

- o Lack of Disk Space
- o System Coldstarts
- o System Upgrades
- o Printer Failures

The total system downtime was approximately 36 person hours.

V. SUMMARY OF HOURS

This is a summary of the hours worked under the terms of the contract dated December 28th, 1987.

Consultant	Previous YTD (hours)	Current Period (hours)	Total YTD (hours)
[redacted]	711.8	203.1	914.9
Donald Buresh	495.5	257.5	753.0
[redacted]	110.5	0.0	110.5
[redacted]	651.2	254.4	905.6
Total	1,969.0	715.0	2,684.0

Total Hours in Contract:	4,182.0	(100.0%)
Total Hours Previous YTD:	- 1,969.0	(47.1%)
Total Hours Current Period:	- 715.0	(17.1%)
Adjustment Because of Billing Rate Differences	- 17.4	(0.4%)
Total Hours Remaining:	1,480.6	(35.4%)

APPENDIX B TYPICAL BASE LEVEL PROJECTION REPORT

TO: [redacted]

FROM: Donald Buresh

CC: [redacted]

DATE: 02-MAY-1988

SUBJECT: BL2, BL6, BL7 and BL8 Projections

Attached is an estimate of the person hours that I expect will be needed to complete BL2, BL6, BL7 and BL8 testing of COBOL85. It demonstrates the need to monitor the coverage for various base levels.

Donald Buresh

B L 2 , B L 6 , B L 7 A N D B L 8 E S T I M A T E S

I. INTRODUCTION

This report updates the projections contained in my 23-MARCH-1988 memo. The new estimates result from completing Milestone-7. Milestone-8 becomes:

Milestone-8 Status Codes and Features	Approx Test Cases New	Approx Test Cases Old	Expected Level of Effort (Person Hours) New	Expected Level of Effort (Person Hours) Old
SC44	30	18	152	118
Indirect Errors and Non-Errors	50	30	252	220
Miscellaneous Features				
OPEN OUTPUT	8	4	40	24
STOP RUN/Close All Files	3	4	16	24
-FILE_ASSIGN	4	2	20	12
	95	58	480	398

To obtain balanced milestones, I moved STANDARD-2 to Milestone-9 which is now:

Milestone-9 Status Codes and Features	Approx Test Cases New	Approx Test Cases Old	Expected Level of Effort (Person Hours) New	Expected Level of Effort (Person Hours) Old
STANDARD-2				
Program Collating Seq.	4	3	24	24
Sort/Merge Coll. Seq.	2	2	12	12
CODE-SET	3	4	18	24
SC39				
Organization Conflict	6	6	32	36
CODE-SET	0	4	0	24
Record Type Conflict	8	8	40	48
Record Size Conflict				
Sequential Organization	11	11	56	68
Relative Organization	11	11	56	68
Indexed Organization	11	11	56	68
Blocking Factors	1	1	4	6
Indexed File Attributes	6	6	32	36
Invalid Magnetic Tape Spec	5	5	24	32
Invalid Assigned Device	4	4	20	24
RBF File	4	4	20	24
No Support for MIDASPLUS				
Variable Length Records	1	1	4	6
Maximum Tape Block Size	1	1	4	6
Maximum Tape Record Size	1	1	4	6
	79	83	406	512

The numbers to the left of a vertical bar are the current estimates, while to the right of a vertical bar are previous projections.

II. ANALYSIS OF MILESTONE-7

In Milestone-7 there were 129 approved test cases and we averaged 25.8 approved test cases per week. In my previous report, I assumed a 20% increase in productivity or 22.3 approved test cases per week. I also stated that QA and Prime together could achieve 6.0 person hours per approved test case. Since we expended 662.1 person hours in Milestone-7, QA averaged 5.1 person hours per approved test case.

However, note that the mini-specs for SC48 and SC49 were approved prior to the beginning of Milestone-7. If I ignore SC48 and SC49 as well as the 60 person hours needed to generate my 23-MARCH-1988 memo, I calculate 5.0 person hours per approved test case. Since Milestone-6 averaged 7.25 person hours per apporved test case, this translates into a 45% increase in productivity which is more than twice what I predicted.

Because we expended 108.2 person hours in the last week of Milestone-7, we averaged 132.5 person hours per week. If I ignore the last week of Milestone-7, we averaged 138.5 person hours per week. This means that it is reasonable to assume that QA will expend 135.0 person hours per week.

From this information, I conclude that the productivity estimates in my 23-MARCH-1988 memo were accurate, if not conservative.

III. POSSIBLE IMPROVEMENTS

In my last report, I mentioned following equation:

$$CONT = RATE \times AVETC \times COVR \qquad \text{Eq. (1)}$$

where

CONT = The amount of the contract in dollars
RATE = The amount paid to us in dollars per hour
AVETC = The average number of hours per approved test case
COVR = The number of approved test cases (coverage)

For confidentiality reasons, dollars are converted into hours. Eq.(1) then becomes:

$$HOURS = AVETC \times COVR \qquad \text{Eq. (2)}$$

where

HOURS = The number of hours in the contract

Note that Eq. (2) is also valid for a specific milestone or a given base level.

On page 3 of my 23-MARCH-1988 memo, I listed following ten (10) activities that must be performed by both Prime and QA to achieve an approved test case.

- o Analysis, specification and tracking of a test case by QA
- o Review of the analysis and specification by Prime
- o Revision of the analysis and specification by QA
- o Approval of the analysis an specification by Prime

- o Coding and debugging of a test case by QA
- o Review of a test case by QA
- o Review of a test case by Prime
- o Revision of a test case by QA
- o Approval of a test case by Prime

- o Integration of a test case into Prime's pre-existing COBOL test bed by QA

The two (2) blank lines mark the end of a cycle in the processs. The first cycle deals with the analysis and specification of a test case and occurs 3 times on the average. The second cycle focuses on generating a test case and usually happens 2 times. The third cycle is not really a cycle at all. It is the integration of a test case and is a one-time event.

AVETC is the sum of the number of person hours for each of the activites that are performed by QA in the above list. Any increase in productivity comes from either:

- o Removing some Prime and QA activities, or

- o Streamlining QA activities.

Given the requirements of the contract, it is highly unlikely that Prime will significantly alter the above process. Furthermore, it appears that any future increases in productivity enjoyed by QA can be attributed to white noise. This means that 5.0 person hours per approved test case seems to be more or less constant.

On the other hand, the number of test cases (COVR) is a parameter which is controlled exclusively by Prime, even though QA provides Prime with an initial estimate. If COVR increases, HOURS increases linearly and Eq. (2) becomes:

$$HOURS = 5.0 \times COVR \qquad \text{Eq. (3)}$$

Since there are a fixed number of person hours in the contract (4,182), it is important that Prime control the number of test cases (COVR) per milestone and per base level.

IV. ESTIMATES OF THE FUTURE

In this section, I want to discuss my perceptions of the future of the contract. I incorporated our current productivity of 5.0 person hours per approved test case. I also recently received BL7 development estimates from Prime, and I want to include this information in this report. The figures that Prime gave me are 250 to 300 person days. I converted person days for Prime employees by multiplying by 7.5 person hours per person day. I did not include BL9 estimates because I do not possess a a development schedule for the base level.

Table 1. estimates the number of person hours needed to complete Milestone-8, Milestone-9, BL6, BL7 and BL8. The numbers in brackets denote the percentages of the current contract (4,182 person hours) consumed by an activity. The figures in parenthesisies specify negative numbers. I assumed that the standard deviation is 20% of the expected value.

In lines (5) through (7), I employed the factor 0.5. I derived this number by dividing 2855.0 estimated person hours to test BL2, assuming 5.0 person hours per approved test case, by 5,850.0 which is the estimated number of person hours (156 person weeks times 37.5 person hours per person week) expended by Prime employees to write and debug BL2. This later value I obtained from Prime.

The table indicates that there are enough person hours in the current contract to complete BL6, BL7 or BL8. Note that Table 1. shows that BL6 and BL8 require the least number of person hours of any two base levels. Although I expect to complete BL7 and BL8 in the current contract, but my confidence level is only 50%. To finish all three base levels, I project that we will expend 705 person hours beyond the current contract or 1,291 person hours at the 90% confidence level.

Table 2. translates Table 1. into person weeks, where a person week consists of 135 person hours. The ending dates for each line number are contained in brackets, where the first date (24-APRIL-88) is the beginning of Milestone-8. Each entry in Table 2. is rounded to near person day. The table also assumes that the standard deviation is 20% of the expected value. As before, the multiplicative factor used in lines (5) through (7) is 0.5.

The critical date in Table 2. is 16-AUG-88, and is the expected termination date of the current contract. The dates in Table 2. that are beyond 16-AUG-88 assume that the current contract will be extended to complete the specified base level or series of base levels. Note that I expect QA to complete BL6, BL7 and BL8 by 22-SEP-88 and on 22-OCT-88 at the 90% confidence level.

TABLE 1: PROJECTED PERSON HOURS FOR BL2, BL6, BL7 AND BL8 ASSUMING QA TIME IS 0.5 OF DEVELOPMENT TIME

Activity	Expected Value (Person Hours)	90% Confidence Level (Person Hours)
(1) Time Remaining in the Contract	2,213 [53%]	2,213 [53%]
(2) Generate Approved Test Cases for Milestone-8	480 [11%]	576 [14%]
(3) Generate Approved Test Cases for Milestone-9	406 [10%]	488 [12%]
(4) Time Remaining after Completing BL2 (4)=(1)-(2)-(3)	1,327 [32%]	1,149 [27%]
(5) Time to Test BL6 (5)=0.5 * 1,388 p.h.	696	836
(6) Time to Test BL7 (6)=0.5 * 1,875 p.h.	940	1,128
(7) Time to Test BL8 (7)=0.5 * 788 p.h.	396	476
(8) Time Remaining after Completing just BL6 (8)=(4)-(5)	631 [15%]	313 [7%]
(9) Time Remaining after Completing just BL7 (9)=(4)-(6)	387 [9%]	21 [1%]
(10) Time Remaining after Completing just BL8 (10)=(4)-(7)	931 [22%]	673 [16%]
(11) Time Remaining after Completing BL6 & BL8 (11)=(4)-(5)-(7)	235 [6%]	(163) [4%]
(12) Time Remaining after Completing BL7 & BL8 (12)=(4)-(6)-(7)	(9) [0%]	(455) [11%]
(13) Time Remaining after Completing BL6 & BL7 (13)=(4)-(5)-(6)	(309) [7%]	(815) [19%]
(14) Time Remaining after Completing BL6, BL7 & BL8 (14)=(4)-(5)-(6)-(7)	(705) [17%]	(1,291) [31%]

TABLE 2: ESTIMATED WEEKS FOR TESTING BL2, BL6, BL7 AND BL8
ASSUMING QA TIME IS 0.5 OF DEVELOPMENT TIME

Activity	Expected Value (Actual Weeks)		90% Confidence Level (Actual Weeks)	
(1) Time Remaining in the Contract	16.4	[24-APR-88]	16.4	[24-APR-88]
(2) Generate Approved Test Cases for Milestone-8	3.6	[18-MAY-88]	4.2	[23-MAY-88]
(3) Generate Approved Test Cases for Milestone-9	3.0	[08-JUN-88]	3.6	[16-JUN-88]
(4) Time Remaining after Completing BL2 (4)=(1)-(2)-(3)	9.8	[16-AUG-88]	8.4	[16-AUG-88]
(5) Time to Test BL6 (5)=[0.8 * 1,388]/135	5.2		6.2	
(6) Time to Test BL7 (6)=[0.8 * 1,875]/135	7.0		8.4	
(7) Time to Test BL8 (7)=[0.8 * 788]/135	3.0		3.6	
(8) Time Remaining after Completing just BL6 (8)=(4)-(5)	4.6	[11-JUL-88]	2.4	[30-JUL-88]
(9) Time Remaining after Completing just BL7 (9)=(4)-(6)	2.8	[27-JUL-88]	0.2	[15-AUG-88]
(10) Time Remaining after Completing just BL8 (10)=(4)-(7)	6.8	[22-JUN-88]	5.0	[12-JUL-88]
(11) Time Remaining after Completing BL6 & BL8 (11)=(4)-(5)-(7)	1.8	[04-AUG-88]	(1.2)	[24-AUG-88]
(12) Time Remaining after Completing BL7 & BL8 (12)=(4)-(6)-(7)	0.0	[16-AUG-88]	(3.4)	[08-SEP-88]
(13) Time Remaining after Completing BL6 & BL7 (13)=(4)-(5)-(6)	(2.2)	[01-SEP-88]	(6.0)	[27-SEP-88]
(14) Time Remaining after Completing BL6, BL7 & BL8 (14)=(4)-(5)-(6)-(7)	(5.2)	[22-SEP-88]	(9.6)	[22-OCT-88]

TABLE 3: PROJECTED PERSON HOURS FOR BL2, BL6, BL7 AND BL8 ASSUMING QA TIME IS 0.8 OF DEVELOPMENT TIME

Activity	Expected Value (Person Hours)	90% Confidence Level (Person Hours)
(1) Time Remaining in the Contract	2,213 [53%]	2,213 [53%]
(2) Generate Approved Test Cases for Milestone-8	480 [11%]	576 [14%]
(3) Generate Approved Test Cases for Milestone-9	406 [10%]	488 [12%]
(4) Time Remaining after Completing BL2 (4)=(1)-(2)-(3)	1,327 [32%]	1,149 [27%]
(5) Time to Test BL6 (5)=0.8 * 1,388 p.h.	1,112	1,336
(6) Time to Test BL7 (6)=0.8 * 1,875 p.h.	1,500	1,800
(7) Time to Test BL8 (7)=0.8 * 788 p.h.	632	760
(8) Time Remaining after Completing just BL6 (8)=(4)-(5)	215 [5%]	(187) [4%]
(9) Time Remaining after Completing just BL7 (9)=(4)-(6)	(173) [4%]	(651) [16%]
(10) Time Remaining after Completing just BL8 (10)=(4)-(7)	695 [17%]	389 [9%]
(11) Time Remaining after Completing BL6 & BL8 (11)=(4)-(5)-(7)	(417) [10%]	(947) [23%]
(12) Time Remaining after Completing BL7 & BL8 (12)=(4)-(6)-(7)	(805) [19%]	(1,411) [34%]
(13) Time Remaining after Completing BL6 & BL7 (13)=(4)-(5)-(6)	(1,285) [31%]	(1,987) [48%]
(14) Time Remaining after Completing BL6, BL7 & BL8 (14)=(4)-(5)-(6)-(7)	(1,917) [46%]	(2,747) [66%]

TABLE 4: ESTIMATED WEEKS FOR TESTING BL2, BL6, BL7 AND BL8
ASSUMING QA TIME IS 0.8 OF DEVELOPMENT TIME

Activity	Expected Value (Actual Weeks)	90% Confidence Level (Actual Weeks)
(1) Time Remaining in the Contract	16.4 [24-APR-88]	16.4 [24-APR-88]
(2) Generate Approved Test Cases for Milestone-8	3.6 [18-MAY-88]	4.2 [23-MAY-88]
(3) Generate Approved Test Cases for Milestone-9	3.0 [08-JUN-88]	3.6 [16-JUN-88]
(4) Time Remaining after Completing BL2 (4)=(1)-(2)-(3)	9.8 [16-AUG-88]	8.4 [16-AUG-88]
(5) Time to Test BL6 (5)=[0.8 * 1,388]/135	8.2	9.8
(6) Time to Test BL7 (6)=[0.8 * 1,875]/135	11.2	13.4
(7) Time to Test BL8 (7)=[0.8 * 788]/135	4.6	5.6
(8) Time Remaining after Completing just BL6 (8)=(4)-(5)	1.6 [04-AUG-88]	(1.4) [24-AUG-88]
(9) Time Remaining after Completing just BL7 (9)=(4)-(6)	(1.2) [25-AUG-88]	(4.8) [19-SEP-88]
(10) Time Remaining after Completing just BL8 (10)=(4)-(7)	5.2 [11-JUL-88]	2.8 [26-JUL-88]
(11) Time Remaining after Completing BL6 & BL8 (11)=(4)-(5)-(7)	(3.0) [06-SEP-88]	(7.0) [03-OCT-88]
(12) Time Remaining after Completing BL7 & BL8 (12)=(4)-(6)-(7)	(6.0) [27-SEP-88]	(10.4) [20-OCT-88]
(13) Time Remaining after Completing BL6 & BL7 (13)=(4)-(5)-(6)	(9.6) [22-OCT-88]	(14.8) [19-NOV-88]
(14) Time Remaining after Completing BL6, BL7 & BL8 (14)=(4)-(5)-(6)-(7)	(14.2) [23-NOV-88]	(20.4) [04-JAN-89]

In the Milestone-7 report, I gave the percent ratio of expected to the actual number of approved test cases. For the test cases defined in the mini-specs, Prime requested 64% more test cases than I projected. Because the mini-specs for SC48 and SC49 were approved prior to the beginning of Milestone-7, the percentage dropped to 46% for approved test cases. Since this means that Prime will probably demand significantly more than what I project, I can adjust the multiplicative constant to 0.8 = 1.64 * 0.5 and estimate the effect on the QA effort.

Given a 64% increase in the number of test cases for BL6, B7 and BL8, I developed Tables 3. and 4. The first conclusion that I draw from Table 3. is that BL6 or BL8 are the only base levels that QA can complete in the current contract. It does not appear possible to complete BL7 or any combination of base levels. Furthermore, I see that an increase of 64% more test cases means that I expect the QA test effort to take 9.0 weeks longer than the estimates in Tables 1. and 2. This means that my estimates are quite sensitive to the amount of coverage demanded by Prime. The more test cases requested by Prime, the longer it will take QA to complete a base level.

I recommend that Prime carefully monitor the number of test cases they require over and above my estimates in Tables 1. and 2. This will ensure that the QA effort is consistent with the overall COBOL85 project estimates.

APPENDIX C: PRODUCTIVITY MEASURES REPORT FOR THE FIRST THREE MILESTONES

TO: [redacted]

FROM: Donald Buresh

CC: [redacted]

DATE: 06-MAY-1988

SUBJECT: Base Level I Productivity Measures

The purpose of this memo is to analyze the productivity of project in BL1. The chart below describes the effort in BL1.

	Milestone II	Milestone III	Total
BL1 Test Cases	87	71	158
BL2 Test Cases	0	32	32
Hours Expended for BL1	814.6	540.7	1,355.3
Hours Expended for BL2	0.0	243.7	243.7
Hours Expended for Milestone	814.6	784.4	1,599.0

Note that no test cases were written in Milestone I. Furthermore, observe that the hours expended for BL1 and BL2 in Milestone III were linearly interpolated based on the number of test cases in each base level.

From the table above, the productivity for BL1 in Milestone II was 9.4 person hours per approved test case, while the productivity for BL1 in Milestone III was 7.6 person hours per approved test case. The overall productivity for BL1 was 8.6 person hours per approved test case.

I did not calculate the productivity for BL2 prior to my arrival for the following reasons:

- Milestone III mixed base levels
- Milestone IV contained personnel changes
- Milestone V possessed approved test cases as well as test case in progress
- The hourly rate for the contract changed between Milestones IV and V

Since our productivity for Milestone VI and VII was 7.25 and 5.0 person hours per approved test case respectively, you see that we are becoming significantly more efficient over time. By multiplying these productivity measures by the appropriate hourly rate, I calculate the unit cost per test case for BL1 and Milestones VI and VII to be $369.80, $333.50 and $230.00 respectively.

I trust that this information that you need for the upcoming negotiations with [redacted] and Prime Computer, Inc.

REFERENCES

1) JAMES R. EVANS, & WILLIAM M. LINDSAY, THE MANAGEMENT AND CONTROL OF QUALITY, (South-Western College Publishing 4th ed. 1999).
2) MARY WALTON, THE DEMING MANAGEMENT METHOD (Perigree Books, 1986).
3) ROGER G. SCHROEDER, *OPERATIONS MANAGEMENT: DECISION MAKING IN THE OPERATIONAL FUNCTION* (McGraw-Hill, Inc.1993).
4) SHOJI SHIBA, ALAN GRAHAM, A. & DAVID WALDEN, A NEW AMERICAN TQM: FOUR PRACTICAL REVOLUTIONS IN MANAGEMENT (Productivity Press 1993).
5) T. S. Elliot, The Hollow Men, *Owl Eyes* (1925), *available at* https://www.owleyes.org/text/the-hollow-men/read/text-of-the-poem#root-42.
6) W. EDWARDS DEMING, OUT OF THE CRISIS (Massachusetts Institute of Technology, Center for Advanced Educational Services 1982).