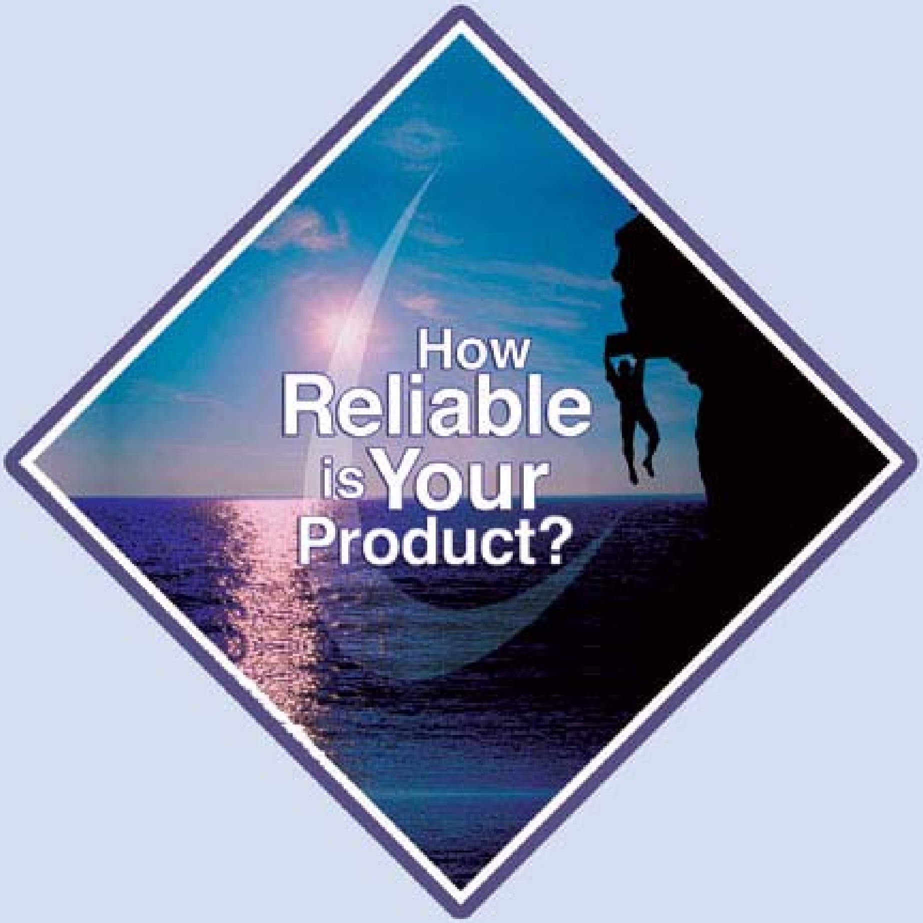


How Reliable is Your Product?

50 WAYS TO IMPROVE PRODUCT RELIABILITY

MIKE SILVERMAN

“How Reliable is Your Product?” Book Excerpt

50 Ways to Improve Product Reliability

By Mike Silverman

Foreword by Patrick O'Connor

Super Star press

E-mail: info@superstarpress.com
20660 Stevens Creek Blvd., Suite 210
Cupertino, CA 95014

BOOK EXCERPT Table of Contents

Contents

NOTE: This is the Table of Contents (TOC) from the book for your reference. The eBook TOC (below) differs in page count from the tradebook TOC.

Foreword by Patrick O'Connor

Modern engineering products, from individual components to large systems, must be designed, developed, and manufactured to be reliable in use. Designs must be robust in relation to the stresses and other factors that could cause damage or deterioration in transport, storage, use, and maintenance. Product development must include testing to ensure that this is achieved and to show up weaknesses for correction. The manufacturing processes must be performed correctly and with the minimum of variation. All of these aspects impact the costs of design, development, manufacture, and use, or, as they are often called, the product's life cycle costs. The challenge of modern competitive engineering is to ensure that life cycle costs are minimized while achieving requirements for performance and time to market. If the market for the product is competitive, improved reliability can generate very strong competitive advantages, as well as cost savings to manufacturers and to users. Today, this message is well understood by most engineering companies that face competitive pressures.

The customers for major systems, particularly the U.S. military, drove the quality and reliability methods that were developed in the West from the 1950s onwards. They reacted to perceived low achievement by the imposition of standards and procedures. The methods included formal systems for quality and reliability management (MIL-Q-9858 and MIL-STD-758) and methods for predicting and measuring reliability (MIL-STD-721, MIL-HDBK-217, and MIL-STD-781). MIL-Q-9858 was the model for the international standard on quality systems (ISO9000). The methods for quantifying reliability were similarly developed and applied to other types of products and have been incorporated into other standards such as ISO60300. The application of these approaches has been controversial and not always effective.

In contrast, the Japanese quality movement that began in the 1950s was led by an industry that learned how manufacturing quality provided the key to greatly increased productivity and competitiveness, principally in commercial and consumer markets. The methods that they applied were based upon understanding of the causes of variation and failures, as well as continuous improvements through the application of process controls and motivation and management of people at work. It is one of history's ironies that the foremost teachers of these ideas were Americans, notably

P. Drucker, W.A. Shewhart, W.E. Deming, and J.R Juran. The Japanese also applied methods for design for reliability, notably Quality Function Deployment (QFD) and failure modes and effects analysis (FMEA).

By the turn of the century, methods of design for reliability and for manufacturing quality excellence had become refined. Most of the U.S. military standards were discontinued. More practical and effective methods were applied almost universally, particularly by industries whose products faced international competition or other drivers, particularly high costs of failures or strict customer requirements. However, some still cling to unrealistic mathematical precision for predicting and measuring reliability, as well as to bureaucratic approaches to quality management.

In the same time frame, there have been improvements in design capabilities with advances in computer-aided engineering, as well as in materials and in manufacturing processes. We have seen dramatic improvements in the reliability of products as diverse as automobiles, telecommunications, domestic equipment, and spacecraft. How many readers have experienced a failure of a microprocessor or an automobile engine?

I am pleased to endorse and recommend this new book. Mike Silverman presents a wealth of practical, experienced-based wisdom in a way that is easy to read and apply. He has avoided detailed descriptions of methods, emphasizing instead the management and team aspects of applying cost-effective reliability improvement tools in ways that work.

The main methods he covers include reliability planning, design techniques (FMEA, fault tree analysis), test—particularly highly accelerated life test (HALT), and design of experiments, as well as methods for reliability prediction, stress derating, vendor reliability, failure reporting and analysis, and others. The whole product life cycle is considered, from initial design through prototype test, manufacturing, and field service, to obsolescence. He emphasizes the need for integration of reliability efforts to ensure their effective application. The fifty chapters all include brief case histories that illustrate this.

I recommend the book as an excellent guide for engineering project management and their teams, as well as for reliability specialists. It demystifies the sometimes difficult methods and helps specialists to communicate with managers, designers, and other engineers. It will make your products more reliable.

Patrick O'Connor
November 2010

Why Am I Writing This Book?

I've read many reliability and quality textbooks, and very few approach reliability from the practical perspective. Instead, these books are filled with theory and formulas. However, many engineers are starting with almost no knowledge on the subject of reliability; they are in need of some basic education, but even more, they need the benefit of some practical experience and guidance. I wrote this book as a helpful guide, and I targeted the book at engineering professionals around the world in need of a practical guide to reliability.

I wrote this book based on my 25 years' practicing reliability, including 10 years running a reliability test lab and 10 years running a reliability consulting firm called Ops A La Carte®. I started Ops A La Carte® because I saw the need to teach and help companies develop reliability programs. Most engineers I come across know basic concepts and have their favorite reliability techniques, but few understand how to put this into an overall reliability program.

Ops A La Carte® has worked with over 500 different companies in over 100 different industries in 30 different countries, so we have the ability to provide guidance from the experiential point of view. When I use the collective term "we" in this book, I am referring to an experience we have had at Ops A La Carte®.

Just like any other discipline, there is no substitute for experience. Book knowledge is a good start, but until you are working on a design program or faced with a particular failure situation, you may not know what to do, or you may panic and resort to ineffective techniques you used in the past. In this book, I will show you different techniques and give you real-life situations that we faced and how we used particular techniques to solve problems.

I saw a movie recently called *Eagle Eye* that is quite applicable to reliability. The movie starts with the discovery of a possible terrorist plot. The Joint Chiefs of Staff of the United States consult their new supercomputer "Eagle Eye" to determine the probability that the terrorist plot is real. The supercomputer comes back with a probability of 39%. The commander in charge responded by saying, "39% and probability don't belong in the same sentence." (So true…and very appropriate for reliability as well.) In the next scene, the Joint Chiefs consult with the President, and by this time, "Eagle

Eye" collects a bit more information and raises its probability to 51%. The President then decides to take action based on this and authorizes an attack on the alleged terrorist group. What is that really telling us? In fact, a probability of 51% means that there is a 49% chance that the conclusion is incorrect based on the data.

Likewise, with reliability tests, you need to make decisions based on test data from a sample of the population. You will never have enough data to be 100% certain of any decision, so you should gain as much confidence as you can with the time and money that you have. That is the art of reliability testing.

I structured the book in 50 easy-to-read chapters. Each chapter has some background on the reliability technique, its usefulness, and in some cases, its limitations. In addition, when applicable, I compare the technique in question to other techniques to show you when to use which technique. Starting in Chapter 3, I introduce the topic of Reliability Integration, and for each chapter onwards, I comment on how you can use the concept of Reliability Integration with that particular technique. I will talk a lot about Reliability Integration. It is one of the most valuable takeaways from this book.

In each chapter, I will provide one or more case studies from clients we have worked with and discuss how we utilized the specific technique in question. I didn't use the names of people or companies, but all of the case studies are real.

Tips on how to best use this book:

- If a phrase is highlighted in bold italics, that means the term is a main technique of that chapter and is in the table of contents as well as the index. I will also capitalize the phrase throughout the rest of the book as an indication that it is an important technique. For all other important terms, check the index for other places I have used the same term.
- I included a guide to acronyms. The field of reliability uses a lot of acronyms and I know how frustrating it can be reading a book filled with them.
- I included a glossary of terms.

If you feel I missed something or you have information to add to a particular topic, I'd love to hear from you. I hope you enjoy my book.

Chapter

3 Reliability Integration Provides Integrity

Reliability Integration is the process of seamlessly and cohesively integrating reliability techniques together to maximize reliability at the lowest possible cost. What this means is you should think of your reliability program as a set of techniques that are used together rather than just a bunch of individual activities.

You are building a system, and a system is made up of different components and assemblies; there are different disciplines involved (some of the main disciplines are electrical, mechanical, software, firmware, optical, and chemical). All of the individual pieces make up the system, so don't forget about the interactions, and make sure that you think of the reliability from a system perspective. In Figure 3.1, we illustrate this point using the disciplines of electrical, mechanical, and software.

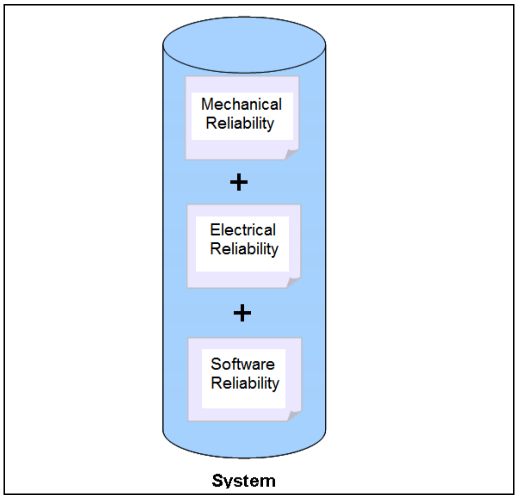


Figure 3.1: System View of Reliability

This is especially true of software versus hardware disciplines. Most companies work on Software Reliability and Hardware Reliability separately and don't integrate the two. When failures occur, this then results in finger-pointing rather than synergy.

This is equally true of electrical versus mechanical disciplines. We see more synergy between these two groups during programs than between software and hardware; however, at the beginning, they rarely get together to discuss common reliability goals and how to apportion them down to each major area of the system.

Product development teams view reliability within each of the separate sub-domains of mechanical, electrical, and software issues. Your customers view reliability as a system-level issue, with minimal concern placed on the distinction between mechanical, electrical, and software issues. Your customer wants the whole product and all its parts to work together perfectly. Since the primary measure of reliability is made by your customer and their end users, engineering teams should maintain a balance of both views (system and sub-domain) in order to develop a reliable product.

3.1 Reliability versus Cost

Intuitively, the emphasis in reliability to achieve a reduction in warranty and in-service costs results in some minimal increase in development and manufacturing costs. Use of the proper techniques during the proper life cycle phase will help to minimize total life cycle cost (LCC).

To minimize total LCC, your organization should do two things:

1. Choose the *best techniques* from all of the techniques available, and apply these techniques at the proper phases of the product life cycle.
2. Properly *integrate* these techniques by feeding information between different phases of the product life cycle.

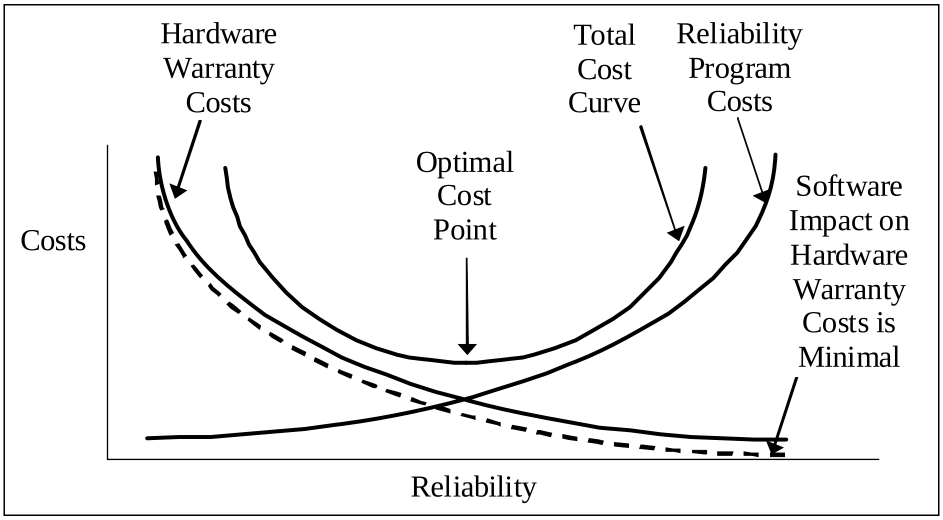


Figure 3.2: System Reliability versus Cost

In Figure 3.2, it is evident that:

1. Program costs go up as you spend more on reliability. At a certain point, you won't get your return on investment (ROI) because the reliability has reached a point where it is becoming increasingly more difficult to improve. That is why it is important to know what the goal is, and it can be just as detrimental to your company to produce a product that is *too* reliable as *not reliable enough*. The product that is too reliable usually comes with increased costs; your customers may not need this level of reliability and will opt for the less expensive product. When was the last time you purchased a $200 blender or toaster?
2. Warranty costs go up as reliability goes down.
3. Software has no associated manufacturing costs (other than perhaps the cost of CDs and manuals and the cost of personnel to test the product in production), so warranty costs and savings are almost entirely allocated to hardware. If there is no cost savings associated with improving Software Reliability, why not leave it as is and focus on improving hardware reliability to save money? You shouldn't do this for two reasons:
    - **a.** Our experience is that for typical systems, software failures outnumber hardware failures by a ratio of 10:1 (see Section 31.1 for more details). Customers buy integrated systems, not just hardware.
    - **b.** The benefits for a Software Reliability program aren't in direct cost savings. Instead, the benefits are in:

i. Increased software/firmware staff availability with reduced operational schedules, resulting in fewer corrective maintenance events.

ii. Increased customer goodwill based on improved customer satisfaction.

CASE STUDY: Linking Electrical, Mechanical, and Software Reliability together

We were working with a semiconductor equipment company to help improve their reliability on their next generation product. First, we provided a Design for Reliability (DFR) seminar for each of the three different disciplines—the electrical group, the mechanical group, and the software group. Then, we met with the electrical, mechanical, and software team leads and developed reliability goals. We started with high level system goals and the apportioned the goals down to each subsystem—electrical, mechanical, and software. Each group lead then took the goal for his subsystem and broke it down further within his area. We worked with each group lead to put together a reliability program plan to meet his subsystem goals. We rolled each of these different subsystem plans into an overall reliability plan for the product. We worked with each group lead to ensure he was on track for meeting his subsystem goals throughout the product development process. The end result was that our client was able to achieve reliability goals for each subsystem and for the system as a whole.

Appendix

A Software Reliability Growth

The following section of the book on Software Reliability Growth was provided by Mark Turner of Enecsys. This is referenced from Section 31.3.5 from the main section of this book.

You can measure and manage the reliability growth of any software (or even hardware) development using an appropriate model, of which many exist, some of which are more suitable than others. Perhaps the most suitable for software development is the Rayleigh model.

Software design and development is a continuous process where you provide functionality using source code. Unfortunately, despite the best intentions of engineers, you may introduce defects as you create source code. Therefore, you will benefit from modeling the creation, identification, and elimination of code defects as a function of time.

Throughout the software development process, there are numerous opportunities for you to introduce defects. We provide a typical Rayleigh curve in Figure A.1, which illustrates the defect insertion and discovery process. This shows how you can identify and address defects that you introduce at an earlier phase in the software development. There will come a point in any software development program where you maximize the defect discovery rate, after which you reduce over time the quantity of remaining defects. Here the leading bar graph illustrates code defect insertion, which can often begin at the start of the development project. Because code defects are often related to the amount of engineering effort, the rate at which you introduce them often is directly proportional to that effort.

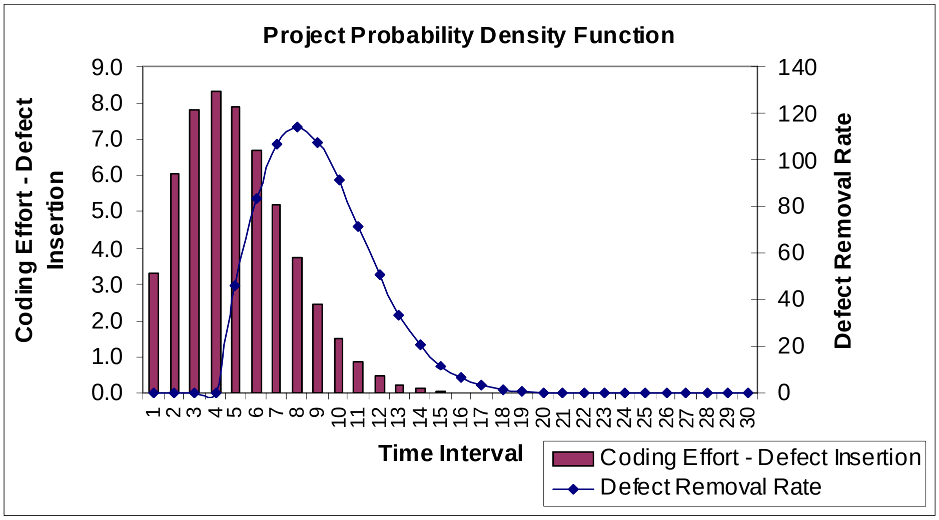


Figure A.1: Rayleigh Estimation Model with Effort and Defect Curves

The lagging curve illustrates the defect removal rate, with problems being addressed at a later date than when you originally inserted them, which can hinder project progress and negatively impact customer satisfaction. You can partially mitigate this by conducting design reviews, code inspections, and early module testing, as these activities will often assist you in discovering inserted defects as early as possible, thus moving the defect discovery and correction curve to the left.

Eventually, the quantity of defects still present in the code will equate to the original reliability target, and as you discover and address further defects, the Software Reliability increases, or grows. You can manage the rate at which you address defects by setting software defect targets. This has to begin by estimating how many defects are likely to occur, then addressing those defects by implementing a Software Reliability growth management program in which you plan and schedule the necessary resource to ensure you achieve your reliability target.

A.1 Implementing the Rayleigh Model

You can use the Rayleigh function to forecast the rate at which you identify defects during the software development program as a function of time. It is a specific instance of one of the models in the Weibull family of reliability models.

This model is particularly suited to Software Reliability modeling as it provides a good representation of the vector sum of a large number of random sources of defects, none of which dominate. The Rayleigh model provides an effective iterative design process in which feedback is inherently part of the solution process, and in fact it closely approximates the actual profile of defect data that you collect during software development programs.

Monitoring software development defect metrics can provide you valuable input into planning engineering and Root Cause Analysis (RCA) efforts, and it helps you to quantify the maturity of the software you are developing. Collecting defect metrics related to engineering effort, project duration, and type over several development projects provides a great opportunity to analyze trends, which can then provide you with more accurate resource predictions for new projects. If you lack such trend data (which typically is a problem when you first deploy the Rayleigh model), then you may have to use industry data as an alternative guide. While this alternative approach may not factor in the abilities of your actual design team, it does at least provide a reasonable estimation to begin with.

After your organization completes multiple projects, you will benefit from reviewing predicted versus actual defect counts, as this enables you to refine the original estimates and improve the model for future development projects. As the project progresses, compare the initial defect count estimates with the quantity of defects you actually address. If you find that the actual defect count is significantly higher than predicted, then the model has generated an early indication that a significant problem may exist. On the other hand, if you find that the actual defect count is significantly less than the initial prediction, then you should confirm that the identification process is indeed sufficient to detect the anticipated defects. Once you confirm this, then you can conclude that your defect insertion rate is actually less than predicted.

In using the Rayleigh model, you should determine the parameters for the total anticipated engineering effort, the total number of defects that you expect to insert into the code, and the time period to reach the peak estimate. Knowing these parameters will enable you to plot the cumulative distribution function (CDF). We've shown an example of a CDF Chart in Figure A.2 and a CDF Table in Figure A.3. The Rayleigh model parameters of Figure A.3 are 55 man-months of engineering effort, 755 inserted defects, and 4 months to reach the peak estimate. For the defect plot, you must define an additional value associated with the estimated lag behind the start of the project effort to account for defect detection and correction effort, which in Figure A.3 is 4 months.

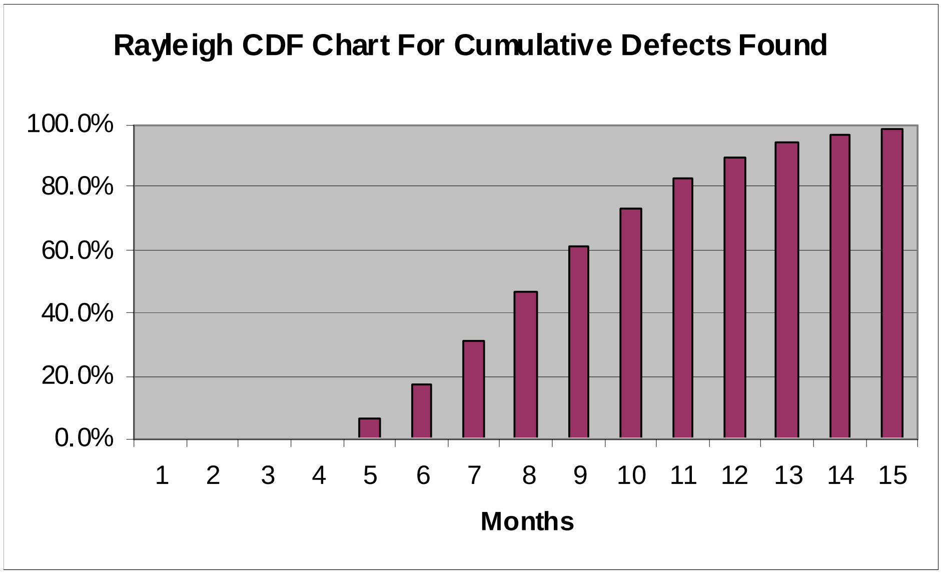
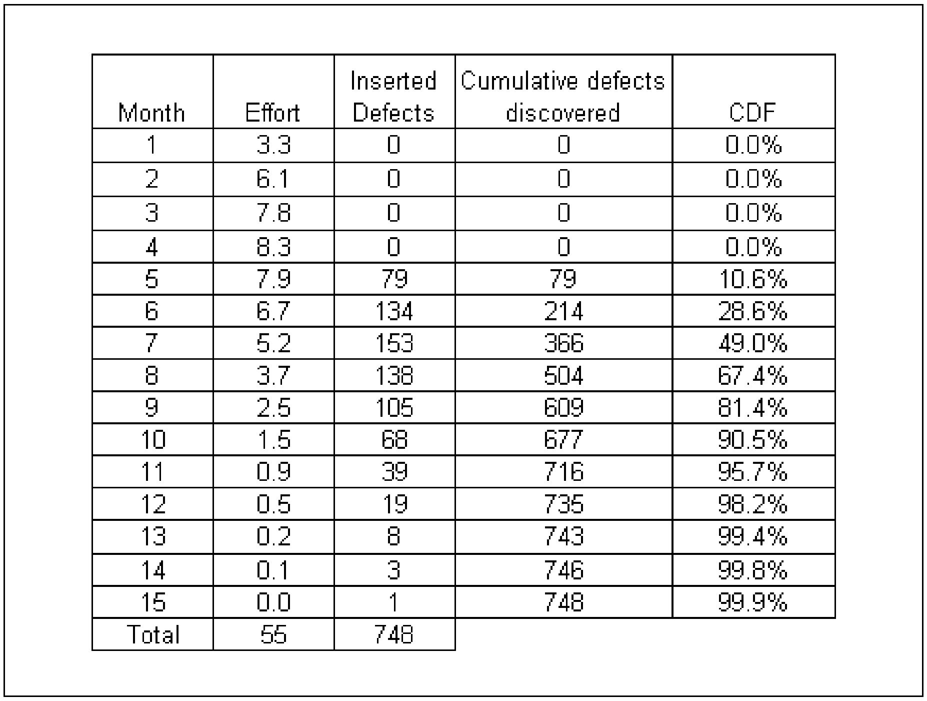


Figure A.2: Rayleigh Cumulative Distribution Function (CDF) Chart

Figures A.2 and A.3 illustrate the relationship between the project effort and the number of defects that you insert into the code, which enables you to make decisions regarding the impact that any code changes are likely to have and in changes to the code delivery date. From this example, you can conclude that a delivery schedule of eight months would be completely unrealistic, as a significant number of defects will still be present in your code, whereas a delivery schedule of twelve to fifteen months is more realistic. Delivery schedules in between require you to make a schedule versus reliability tradeoff.



Month	Effort	Inserted Defects	Cumulative defects discovered	CDF
1	3.3	0	0	0.0%
2	6.1	0	0	0.0%
3	7.8	0	0	0.0%
4	8.3	0	0	0.0%
5	7.9	79	79	10.6%
6	6.7	134	214	28.6%
7	5.2	153	366	49.0%
8	3.7	138	504	67.4%
9	2.5	105	609	81.4%
10	1.5	68	677	90.5%
11	0.9	39	716	95.7%
12	0.5	19	735	98.2%
13	0.2	8	743	99.4%
14	0.1	3	746	99.8%
15	0.0	1	748	99.9%
Total	55	748		

Figure A.3: Rayleigh CDF Table

However, if early delivery is unavoidable, the CDF can aid in planning reliability growth activities and managing customer expectations where multiple deliveries are viable.

About the Author

Mike Silverman

Mike is founder and managing partner at Ops A La Carte LLC®, a professional consulting company that has an intense focus on helping clients with reliability throughout their product life cycle. Mike has over 25 years' experience in reliability engineering, reliability management, and reliability training. He is an experienced leader in reliability improvement through analysis and testing. Through Ops A La Carte®, Mike has had extensive experience as a consultant to high-tech companies. A few of the main industries are aerospace and defense, clean technology, consumer electronics, medical, networking, oil and gas, semiconductor equipment, and telecommunications. Most of the examples in this book have been taken from Mike's experiences.

Mike is an expert in accelerated reliability techniques and owns HALT and HASS Labs®, one of the oldest and most experienced reliability labs in the world. Mike has authored and published 25 papers on reliability techniques and has presented these in countries around the world, including Canada, China, Germany, Japan, Korea, Singapore, Taiwan, and the USA. He has also developed and currently teaches over 30 courses on reliability techniques. Mike has a BS degree in electrical and computer engineering from the University of Colorado at Boulder, and is a Certified Reliability Engineer (CRE) through the American Society for Quality (ASQ). Mike is a member of the American Society for Quality (ASQ), Institute of Electrical and Electronics Engineers (IEEE), Product Realization Group (PRG), Professional and Technical Consultants Association (PATCA), and IEEE Consulting Society. Mike is currently the IEEE Reliability Society Santa Clara Valley Chapter Chair.

You can contact Mike via the Ops A La Carte® website at
http://www.opsalacarte.com.

Getting “How Reliable is Your Product?” (http://happyabout.com/productreliability.php)

“How Reliable is Your Product?” can be purchased as an eBook for $19.95 or tradebook for $44.95 at http://happyabout.com/productreliability.php or at other online and physical book stores.

- Please contact us for quantity discounts at sales@happyabout.info
- If you want to be informed by email of upcoming Happy About® books, please email bookupdate@happyabout.info

Happy About is interested in you if you are an author who would like to submit a non-fiction book proposal or a corporation that would like to have a book written for you. Please contact us by email editorial@happyabout.info or phone (1-408-257-3000).

Other Happy About books available include:

- #LEADERSHIPtweet Book01: **http://www.happyabout.com/thinkaha/leadershiptweet01.php**
- #PARTNER tweet Book01: **http://www.happyabout.com/thinkaha/partnertweet01.php**
- Expert Product Management: **http://www.happyabout.com/expertproductmanagement.php**
- 42 Rules of Product Management: **http://www.happyabout.com/42rules/42rulesproductmanagement.php**
- 42 Rules to Increase Sales Effectiveness: **http://www.happyabout.com/42rules/increasesaleseffectiveness.php**
- 42 Rules for Driving Success With Books: **http://www.happyabout.com/42rules/books-drive-success.php**
- Agile Excellence for Product Managers: **http://www.happyabout.com/agileproductmangers.php**
- The Phenomenal Product Manager: **http://www.happyabout.com/phenomenal-product-manager.php**
- #QUALITYtweet Book01: **http://www.happyabout.com/thinkaha/qualitytweet01.php**
- Scrappy Project Management®: **http://www.happyabout.com/scrappyabout/project-management.php**
- #PROJECT MANAGEMENT tweet Book01: **http://www.happyabout.com/thinkaha/projectmanagementtweet01.php**

The book presents practical examples of what an engineer should and shouldn't do to be successful in developing a reliable product. It is a must read for all engineers and managers who are involved with corporate-wide improvement efforts.
James McLinn, CRE, Fellow ASQ, Manager of Reliability, Beckman-Coulter

Mike takes you on a guided tour of Design for Reliability techniques, which are interwoven with many fascinating and useful case studies that demonstrate how this approach has already benefited many organizations. He has gifted you with the tools to help make your great products better.
Mark Turner, C.R.P., Six Sigma & DFSS Black Belt, Manager of Reliability, Enecsys

If most people who want reliable products would execute even half of what he is espousing, how great it would be for consumers. Instead of going into depth on each topic, Mike stays at a higher level to give a perspective of reliability within the product development lifecycle. Enjoy his book!
Harry McLean, Manager of Reliability, Advanced Energy Inc.
Author of Book *HALT, HASS, and HASA Explained: Accelerated Reliability Techniques*

Throughout his 25-year career, ***Mike Silverman*** has maintained a singular focus on reliability. He is Founder of and Managing Partner at Ops A La Carte, a reliability engineering consultancy that helps customers build end-to-end reliability into their products. He owns and operates HALT and HASS Labs, a reliability laboratory in Northern California that has now tested over 500 products. A Certified Reliability Engineer with over a dozen technical papers under his belt, Mike is currently President of the Silicon Valley IEEE Reliability Society.

DESIGN : PRODUCT : MANAGEMENT &
LEADERSHIP : QUALITY CONTROL

$44.95 US

SUPERSTAR press
http://superstarpress.com