1

The Need for a New Paradigm for Hardware Reliability and Software Quality

1.1 Rapidly Shifting Challenges for Hardware Reliability and Software Quality

Hardware reliability and software quality, why do you need it? The major US car manufacturers saw their dominance eroded by the Japanese automobile manufacturers during the 1970s because the vehicles produced by the big three had significantly more problems. The slow downward market slide of the US automobile industry was predictable when the defect rate of US automobiles was compared with the Japanese automobile industry. In 1981, a Japanese-manufactured automobile averaged 240 defects per 100 cars. The US automobile manufacturers, during the same time period, were manufacturing vehicles with 280–360% more defects per 100 vehicles. General Motors averaged 670 defects per 100 cars, Ford averaged 740 defects per 100 cars, and Chrysler was the highest, with 870 defects per 100 cars.

Much has been written about how this came about and how the US manufacturers began implementing total quality management (TQM), quality circles, continuous improvement, and concurrent engineering to improve their products. Now the US automobile industry produces quality vehicles, and the perception that Japanese vehicles are better has eroded significantly. J.D. Powers and Associates reported in its 1997 model year report that cars and trucks averaged about 100 defects per 100 vehicles. This represented a 22% increase from 1996 and a 100% decrease from 1987. Vehicles such as the GM Saturn and Ford Taurus are a tribute to that success, both in financial terms and in the improved perception that automobile manufacturers in the United States can produce reliable, quality automobiles. Quality programs like TQM have dramatically improved American manufacturing quality. The automotive industry has also benefited from the quality of the components going into automobiles, which is also at a very high quality level. Counterfeit components and counterfeit material is still a major concern for the electronics and automotive industry that requires constant diligence and an effective program to minimize the risk of counterfeit material entering into the production stream.

In the 1970s, the typical automobile warranty was for 12 months or 12 000 miles. In 1997, automobile manufactures were offering 3-year/36 000-mile bumper-to-bumper warranties. Three years later, these same automobile manufacturers were offering 7-year/100 000-mile warranties. Jaguar is now advertising a 7-year/100 000-mile warranty on its used vehicles! BMW has responded with a similar type of program. The reason these manufacturers can offer longer warranty periods is because they

Improving Product Reliability and Software Quality: Strategies, Tools, Process and Implementation,
Second Edition. Mark A. Levin, Ted T. Kalal and Jonathan Rodin.

understand why and how their vehicles are failing and can therefore produce more reliable vehicles.

A 1997 consumer reports survey of 604,000 automobile owners showed a dramatic improvement in the perception of the reliability of US-manufactured automobiles. The improvement by the big three automobile manufacturers did not occur overnight. It was the result of a commitment to provide the necessary resources along with a credible plan for producing reliable vehicles. It was a paradigm change that took years and evolved through many steps.

The process to improve hardware reliability has made significant progress over the past 20 years. If you follow the process outlined, there can be significant improvements to your product reliability. The hardware reliability errors are often the result of either not following or poorly executing the hardware reliability process rather than being a weakness in the reliability process.

Reliability research and development continues in some areas, such as prognostics and health management (PHM). PHM can improve product maintainability, reduce unscheduled downtime, and lower the cost of ownership over the lifetime of a product. PHM uses real-time sensors to monitor the health of a system. The sensor data is then compared to a good set of data to determine if the system is degrading and to estimate the time to failure. PHM strategies are being use in the automotive, aerospace, and other industries.

Even though hardware reliability has improved significantly for many companies, software quality and software security have become a bigger issue. This is partly due to the fact that many of the new products being developed require significantly more software and firmware. For example, McKinsey & Company estimates that over 10% of automobile vehicle content today (2018) is software and that software will reach 30% of vehicle content by 2030 [1]. Many of the hardware products being developed increasingly need software and firmware to function properly. Increasingly, software is also being deployed to manage critical safety and health operations such as robotic surgery and self-driving cars. It is not uncommon for the software development team to be inadequately sized for the staff and skill level needed to support software development. Automation, the Internet of things (IOT), advances in Wi-Fi and Bluetooth, and greater use of the internet to improve customer experience all drive the need for increased software code development and an improved software quality system. The Internet can be used to push out software updates effortlessly to the end user, but a poor software quality process results in injecting more software bugs than it fixes.

The number of software-to-hardware bugs that need to be fixed during product development can be in the order of 50 to 100 : 1. The complexity of software continues to increase along with new software languages and new drivers that create compatibility issues. About 40% of software development cost is to support testing for software verification. Software quality bugs are design faults that need to be identified, prioritized, and fixed. The traditional software quality systems addressed software bugs downstream as part of the final production testing. For many of the products being developed, this is too late in the development process. This is driving the need to improve software quality and the software development process from the perspective of project management, software requirements, and performance. This includes improvements to the software development process and staffing the team with domain experts.

1.2 Gaining Competitive Advantage

Companies successfully competing in the twenty-first century share a common thread. They all produce quality products that meet or exceed customer expectations over time. This may not seem like a revelation, but the process and tools that these companies will use to achieve this success must be new. In some industries, technology moves so fast that customers tend to trade up to the next-generation product before the first model stops performing to specification. This may seem like the ideal environment for a manufacturer because the product life expectations of the consumer are shorter. However, in reality, achieving product reliability with decreasing product development times requires a change in the way we develop products. Platform product development times have shortened to 18 months and their derivatives (product offshoots) have shrunk to 12 months or less. Of course, this is highly dependent on the product complexity and regulatory and safety requirements, but the trend cannot be ignored. Companies pay a heavy price for releasing a product that is "buggy" or unreliable. Satisfied customers are repeat customers. It is a well-known fact that it costs 5—10 times more to acquire new customers than it does to retain existing ones. It doesn't matter whether you are competing on cost or product differentiation; reliable products result in repeat customers and product growth through word of mouth. A faulty product usually results in the customer communicating dissatisfaction to anyone who will listen until the product or service is replaced with a more reliable one.

1.3 Competing in the Next Decade – Winners Will Compete on Reliability

The business practices of the past few decades will not be sufficient to ensure success in the twenty-first century. Through the years, we've learned to master the skill of building quality products. Higher-quality products have resulted in improved profit margins. In fact, consumers make buying decisions based on their perception of which products have better quality when the competing products were of the same approximate price. In the past few decades, reliability was not a deciding factor for most consumers. This is mostly the result of the consumer's lack of knowledge about product quality. However, the average consumer in the twenty-first century will make buying decisions based not only on price and quality but also on the perceived reliability of the product. Consumers make buying decisions based on which product offers the best value. We can define product value as

$$\text{Product value} = \frac{\text{Customer-perceived value}}{\text{Price}}$$

Here the customer-perceived value is related to the quality and reliability of the product. One of the key advantages of implementing reliability throughout the organization and at every phase of the product life is that the product value increases because of an improved customer perception of the value of the product and a lower cost of production. There is a common misperception that implementing reliability delays the product development time and increases the cost of the product (both in material and production costs). The reality is the exact opposite. Products that are more reliable generally have lower production costs. The reason for this is the result of many factors that contribute to reducing product costs and the product development cycle. For example, products that are reliable generally have

- Higher first-pass yield in test,
- Less material scrap,
- Less product rework (which helps to lower product cost and improve product reliability),
- Fewer field failures,
- Reduced warranty costs (this saving can be passed onto the consumer to provide a competitive price advantage),
- Lower risk of recall,
- Superior designs that are easier to manufacture.

Looking back at the definition of what the consumer considers to be of value, it becomes clear that product reliability will increase the perceived product value and lower the cost of production. This is an important fact about product reliability that is often misunderstood.

1.4 Concurrent Engineering

An important ingredient for successful design and implementation of new technologies into manufacturing involves the establishment of concurrent engineering practices. Concurrent engineering is a process used from design concept through product development and into manufacturing in which cross-functional representatives from all relevant departments provide input on key decisions. These decisions have a direct impact on the price, performance, quality, and development time required for the product. The concept was discussed extensively in the 1990s and has resulted in better products, shorter product development times, and greater profits for those who use it. However, the cross-functional teams consisted of marketing, design, test, and manufacturing. The teams did not include a separate representative for reliability, since this was considered part of the design and test engineers' responsibility. (We will see in Part II of this book that the tools used to improve product reliability are unknown to most design and test engineers.)

This convention needs to be changed and a more encompassing version of concurrent engineering must be developed that takes into account the entire product life cycle. The product life cycle approach includes reliability, serviceability, and maintainability inputs that begin in the design concept phase and continue through product development and product life. This cradle-to-grave approach ensures that the lessons learned along the way are captured and incorporated into the next development cycle.

Previous approaches to product development relied heavily on early design for manufacturing (DFM) effort and prototype testing to catch design flaws prior to product release. The problem with this approach is that DFM engineers (being highly skilled in the manufacturing process) primarily ensure that a product is manufacturable and can be rapidly ramped in production to meet market forecasts. Put another way, DFM ensures that the products designed can be ramped in production with ease (high-quality products), but the effort contributes little to product reliability.

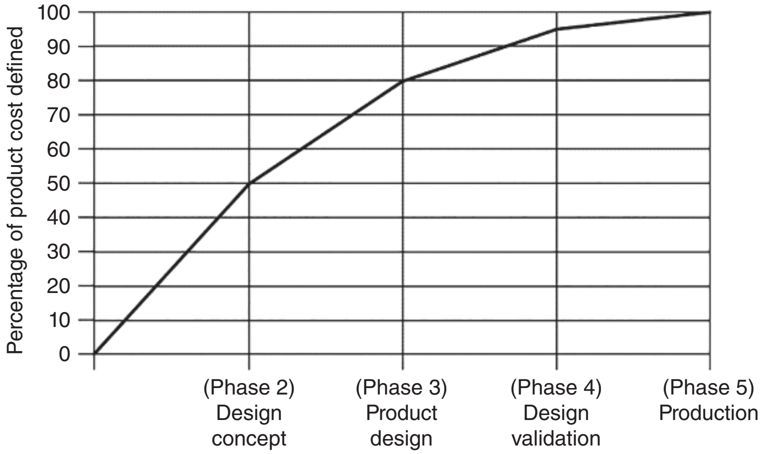

Figure 1.1 Product cost is determined early in development.

Testing performed at the prototype stage will validate product performance to specification prior to engineering release. This does not, however, consider the ability of every product produced to meet specifications in manufacturing. The problem with this approach is that decisions are continuously made in product development that have significant impact on the product performance, reliability, and the ease with which the product can be serviced and maintained. At this stage, decisions need to be made fast, and they should include inputs from everyone affected: marketing, design, test, manufacturing, field service, and reliability.

As stated earlier, it is important to involve all the relevant organizations and support groups early in the product development cycle in order to ensure the lowest product cost and highest product reliability. Design programs including DFM, as well as design for test (DFT), design for reliability (DFR), design for service (DFS), and maintainability must be considered early in the product concept phase. Representatives of each of these functions provide inputs based on guidelines developed from industry standards, lessons learned, intellectual property, and internal process development. These decisions must be made on the basis of facts, not perceptions.

Applying these design guidelines concurrently to product development will continuously reduce cost and cycle time, while also optimizing reliability. Figure 1.1 illustrates how a product life cycle approach to product development will have a direct, positive impact on the cost of the product. Typically, 80% of product cost is committed by the time it goes into prototyping. *Consequently, the greatest opportunity to reduce the cost of a product is in the design phase.* The product life cycle approach addresses all issues that affect the cost, service, reliability, and maintainability of the product for the entire life cycle of the product. These activities include involvement of the entire team on decisions that affect new technologies, packages, processes, and designs, and are based on a cost–benefit analysis, which includes market research, risk, and reliability.

Another driving reason for incorporating reliability as early as possible into product development is the cost of a change based on manpower and capital, when it is made after the design concept phase. Figure 1.1 illustrates how dramatic this impact can be on product cost. The greatest opportunity to reduce cost is in the development and

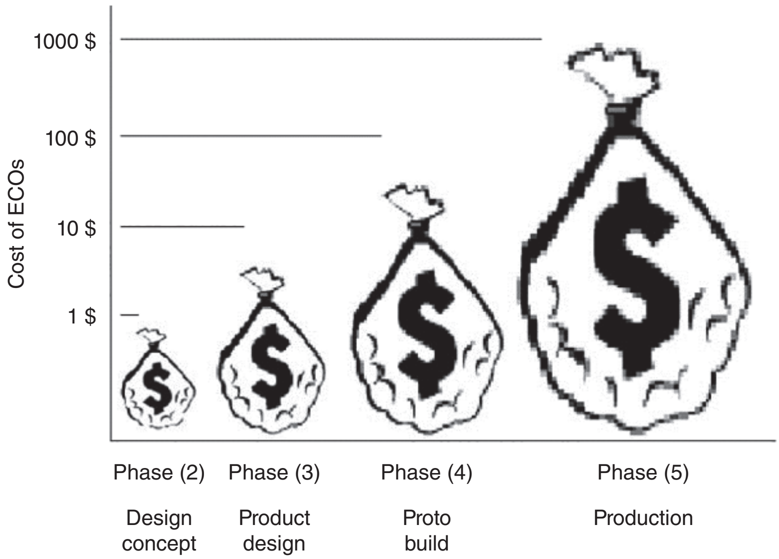

Figure 1.2 Cost to fix a design increases an order of magnitude with each subsequent phase. Source: Courtesy of Teradyne, Inc.

design concept phase, when risk issues relating to technology, components, and processes determine the majority of the product cost. By applying these practices early in the design phase, the cost and labor resources required for implementing engineering changes can be greatly reduced.

This point is illustrated further in Figure 1.2 in which the total cost of an engineering change can increase by several orders of magnitude when it is made late in the product development cycle.

1.5 Reducing the Number of Engineering Change Orders at Product Release

In Part IV, "Reliability Process for Product Development," we will show how using tools such as Highly Accelerated Life Test (HALT™), Highly Accelerated Stress Screening (HASS™), failure modes and effects analysis (FMEA), and risk mitigation early in the product development cycle will reveal hidden problems that are usually not caught until the product has been in production for some time. The product life cycle approach will also reduce the number of engineering change orders (ECOs) at product introduction and increase long-term product reliability. This idea is best illustrated in Figure 1.3 in which a product life cycle approach ensures that the majority of the engineering design changes occur early in the development cycle. This is the best way to reduce the risk of field returns after product release. The graph illustrates how the number of field returns and engineering changes is significantly reduced through early implementation of reliability in the product development cycle.

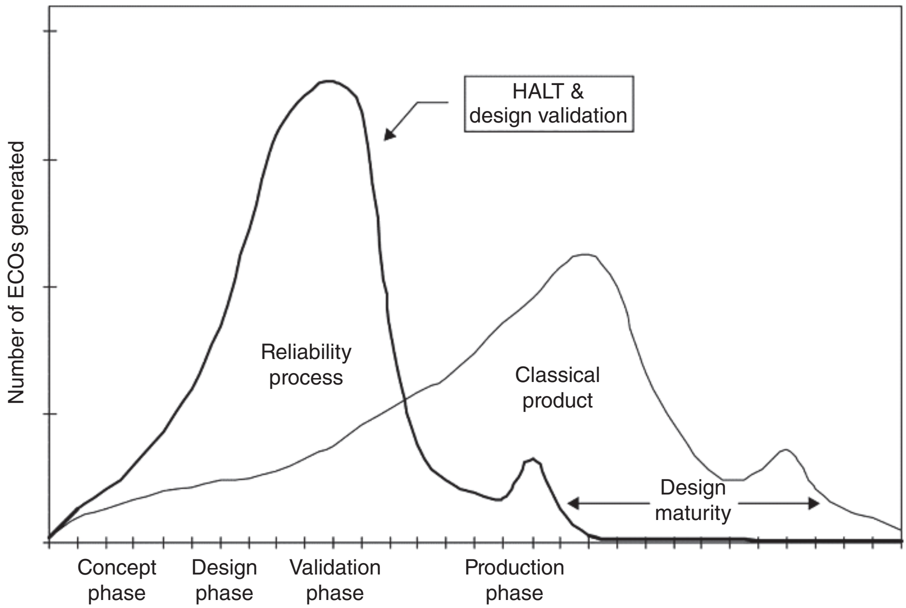

Figure 1.3 The reliability process reduces the number of ECOs required after product release.

1.6 Time-to-Market Advantage

One of the driving forces affecting product reliability is the greatly reduced product development cycles that organizations are facing. Coincidentally, this is also the biggest reason cited for why a product cannot undergo the additional activities required for reliability. But this argument is contrary to what actually happens when reliability is included early in the design concept phase. A major advantage to the implementation of a product life cycle approach to reliability is the reduced development time for product introduction. When time-to-market goals are achieved, the benefits include product name recognition, the ability to set industry standards, recognition as a leader, expansion of the customer base, and the maximization of profits. Using a product life cycle approach, product development time will be significantly reduced, as is shown in Figure 1.4.

Finally, Figure 1.5 illustrates how the timing of product introduction can affect product profitability. Introducing a new product at the same time as the competition will lead to average profits over the life of the product. By releasing a product ahead of the competition, the opportunity for profits increases. Conversely, when releasing a product after the competition, the opportunity for profits is much lower. It is important to point out that getting too far ahead of the market can be undesirable. This point is illustrated in the article "*A Survey of Major Approaches for Accelerating New Product Development*" [2], in which a late entrant in the memory chip market may not receive any profit or even recoup its investment.

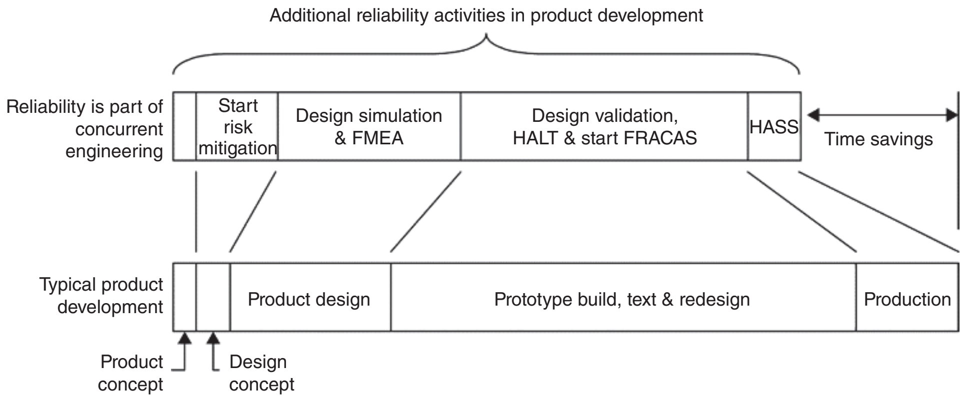

Figure 1.4 Including reliability in concurrent engineering reduces time to market.

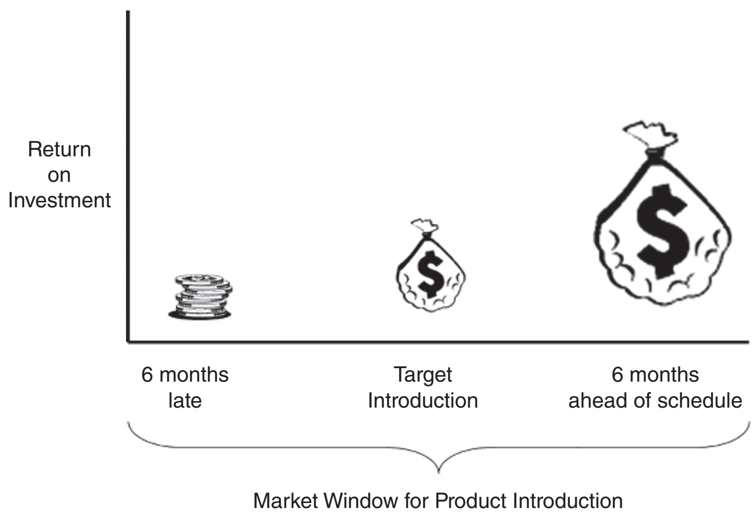

Figure 1.5 Product introduction relative to competitors.

1.7 Accelerating Product Development

There are many ways to accelerate the development of a new product. Product development can be accelerated by simplifying the product design process, improving communications between the cross-functional organizations, implementing an escalation process to resolve conflicts, eliminating unnecessary steps, maintaining development workload at no higher than 85% of capacity, parallel processing as much as possible, and most importantly, eliminating delays.

In today's competitive technology environment, companies can no longer afford to be late with a new product release, especially in a market in which product life continues to decrease. It is important for a company to eliminate the "not invented here" attitude that can often lead to overlooking a simpler solution or a new opportunity.

Another way to simplify manufacturing and to shorten the product development cycle is by standardizing common designs with a modular structure. Hewlett-Packard has been very successful in creating products with modular design. Modularity renders itself to easy upgrades and new design features. In addition, by standardizing certain common features inherent to all products, the development time is greatly reduced. This also eliminates the problem of having many different versions of a common feature. For example, a common circuit like an amplifier with 10 dB of gain could be designed differently by each engineer, but would perform the same function. If the 10-dB amplifier were standardized, then engineering time would not be wasted on "reinventing the wheel" and there would be a high assurance that the new circuit would work.

1.8 Identifying and Managing Risks

The key to any reliability program is the identification of risk. This concept is addressed in great detail in Part IV in which the tools and process for implementing reliability into the product life cycle process are presented. A credible reliability program must focus on the high-risk issues in the project. There will be risk issues at every stage of the product life cycle. Early in the concept phase, decisions are made relating to the features and specifications needed to capture the target market. Marketing uses extensive voice of the customer (VOC) to identify the next high-growth opportunity. These growth opportunities usually involve new technologies. For businesses that compete on the cutting edge of technology, new technologies represent a significant portion of the risk to the program and long-term reliability.

To develop the new platform product, a list of challenges must be devised. Each of these challenges represents risk to the program. To manage the risk, each item should be ranked on the severity of the risk and those items representing the highest risk should be tracked through the program. The role of reliability in the concept phase is to ensure that the risk issues are properly identified. They should be ranked by severity with corrective actions listed so that, when completed, the risk is mitigated. The risk plan needs to include all the functions that are affected in the life cycle of the product. Risk issues that relate to maintenance, manufacturability, design, safety, and environment are included. Unfortunately, these activities are reactive and thus add to the program development time. However, it will be shown later that the net result of these activities is the reduction in product development cost.

There are also proactive reliability activities that occur in advance of product development that help reduce product development time and improve product reliability. An example of a proactive reliability activity to reduce technology risk is the technology roadmap. Early VOC will identify future market needs that require new technologies. Mitigating technology risk issues in advance of need provides the required time to fully mitigate all risk issues. By mitigating technology risk in advance of need, you can gain competitive advantage by being the first to market, capturing a greater market share than would otherwise have been possible.

1.9 ICM, a Process to Mitigate Risk

One of the biggest challenges in any development program is to identify and mitigate the risk issues early in the program. This can be best achieved through a technique called identify, communicate, and mitigate (ICM), which is illustrated in Figure 1.6. The ICM approach is a three-step process to identify significant risk to the program,

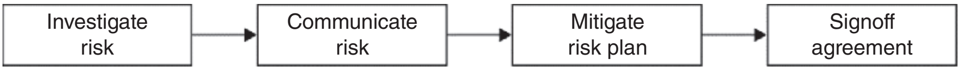

Figure 1.6 The ICM process.

to communicate its impact, and to develop an agreed-upon strategy to mitigate the risk. The ICM approach is an effective way to allocate and utilize limited resources in a way that will most benefit the program goals.

To identify the risk, each functional group reviews the product concepts, designs, and processes. Each group then identifies issues in which the present technology, methods, materials, processes, and tools cannot ensure success. These are the risks. Early risk identification makes the entire team focus on the concept of product reliability at the earliest possible time. All risk issues, no matter how small, need to be identified during this phase of the process. The low-risk issues will not have the same visibility and priority as the high-risk issues. In order to ensure that the major risk issues get completed prior to first customer ship, all risk issues are captured at the earliest possible point, the severity is ranked, and a risk mitigation plan is put in place based on the severity of each issue. Risk identification is a process that must be formalized and documented with the following captured information:

1. Description of the risk
2. Brief description of the activity needed to mitigate risk
3. Impact of risk to program and other functional groups
4. Severity of the risk defined on a scale from insignificant to catastrophic
5. Alternate solutions.

1.10 Software Quality Overview

Delivering high-quality software is critical to the success of a product and company. The lack of quality processes adds a lot of cost to software development organizations. The National Institute of Standards and Technology (NIST) determined in a 2002 study [3] that the cost to software development organizations in the United States due to a lack of adequate quality assurance infrastructure and process was estimated between $22 billion and $59 billion. With the growth of the software industry and inflation, that economic impact is now roughly double that 2002 estimate. This is a heavy productivity burden for a software development group to carry in a competitive marketplace and is a major drain on profitability. In addition to the overhead costs that comes from inadequate quality processes, a product's success often depends on its quality. If the product does not work or is too buggy, it will not succeed.

There is a trend in the industry to deliver more frequent software releases. This can be due either to a need to stay ahead of the competitors in consumer device markets or to the insatiable customer appetite for new features on top of existing hardware. Trying to support low-quality and buggy software slows down the ability to deliver timely software releases.

Fortunately, the process to create high-quality, reliable software is not very complicated. But, that does not mean it is easy. Developing quality software requires discipline in the application of techniques, processes, metrics, and controls. The quality of the software cannot be known unless it is measured. The production of quality software cannot occur unless procedures and tools are used to prevent, discover, and fix defects in the software during the development process. This book provides an introduction to basic metrics, tools, and procedures that can be used to produce high-quality software and to improve software quality over time.

The software described is a data-driven style of software development management. Quantifiable goals are set. Techniques and procedures are deployed throughout the software development process to prevent and find the software defects. Measures are taken and updated throughout the development process to track the current quality level and determine readiness for software release.

There are two overlapping sets of processes described here. The first consists of a set of techniques and metrics for delivering a high-quality software release. The second set of processes consists of methods to improve the software quality capabilities over time, resulting in continuous improvement from release to release.

The general software quality process cycle consists of four steps:

1. Set a quality goal for each software release.
2. Track the quality of the software as early as possible during the development process.
3. Employ techniques and procedures to prevent, detect, and fix defects during the development of a software release.
4. Analyze the quality misses after the release and evolve the processes to improve quality for subsequent software releases.

This book covers many aspects of the software life cycle, processes, methodologies, tools, and metrics that apply to creating and releasing high-quality software. However, this is not a book about software life cycles and processes; there are many excellent books that cover those topics in depth. This book only covers those topics insofar as they apply to producing quality software.

This book contains examples of pseudocode, test plans, and metrics. It should be noted that these examples are for illustrative purposes only. They are not intended to be complete, compilable, or production worthy.

References

1 Burkacky, O., Deichmann, J., Doll, G. et al. (2018). *Rethinking Car Software and Electronics Architecture*. McKinsey & Company.

2 Millson, M., Raj, S.P., and Wilemon, D. (1992). A survey of major approaches for accelerating new product development. *The Journal of Product Innovation Management* 9: 55.

3 Tassey, G. (2002). *The Economic Impact of Inadequate Infrastructure for Software Testing*. National Institute of Standards and Technology.

Further Reading

Hoffman, D.R. (1998). *An Overview of Concurrent Engineering*, 1997 Proceedings Annual Reliability and Maintainability Symposium (1998).