Metrics Guidebook for Integrated Systems and Product Development

Document No.: INCOSE-TP-1995-002-01

Sponsor
INCOSE Measurement Technical Committee

Approval July 25, 1995
INCOSE Technical Board

This document was prepared by the Metrics Working Group of the International Council on Systems Engineering. It has received interim approval of the INCOSE Technical Board.

INCOSE technical documents are developed within the working groups of INCOSE. Members of the working groups serve voluntarily and without compensation. They are not necessarily members of INCOSE. The reports developed within INCOSE represent a consensus of the broad expertise on the subject within INCOSE as well as those activities outside of INCOSE that have expressed an interest in participating in the development of the report.

Comments for revision of INCOSE technical reports are welcome from any interested party, regardless of membership affiliation with INCOSE. Suggestions for change in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on technical reports and requests for interpretations should be addressed to:

National Director, INCOSE
2033 Sixth Avenue #804
Seattle, WA 98121
The authors of this Guidebook

Ann Wilbur  Gayle Towers
Tom Sherman  Dan Yasukawa  Sue Shreve

wish to acknowledge with gratitude the contributions of the individuals and the support of the corporations and organizations below.

Loral Western Development Labs
Loral Space & Range Systems
Lockheed Martin Missiles & Space Company
Applied Materials
San Francisco Bay Area INCOSE Chapter
INCOSE:
Gerry Fisher, Benchmarking Working Group
Art Pyster, Measurement Technical Committee
Table of Contents

1 INTRODUCTION ................................................................................................................................. 1
  1.1 GUIDEBOOK PURPOSE .................................................................................................................. 1
  1.2 GUIDEBOOK BACKGROUND .......................................................................................................... 1
  1.3 GUIDEBOOK OBJECTIVES ............................................................................................................ 3
  1.4 AUDIENCE ..................................................................................................................................... 3
  1.5 HOW TO USE THIS GUIDEBOOK .............................................................................................. 3
  1.6 GUIDEBOOK ORGANIZATION ..................................................................................................... 4

2 MEASUREMENT PROCESS OVERVIEW ............................................................................................. 5
  2.1 UNDERSTANDING METRICS VS. MEASURES ........................................................................... 5
    2.1.1 The Three Great Metrics Controversies: Taxonomy, Systems Engineering (SE) ...
    2.1.2 Attributes of a Good Metric .................................................................................................... 6
    2.1.3 What a Metric is Not .............................................................................................................. 6
  2.2 BENEFITS OF USING METRICS .................................................................................................... 7
    2.2.1 Benefits of Metrics to the Customer ....................................................................................... 8
    2.2.2 Benefits of Metrics to the Performing Organization ............................................................ 8
    2.2.3 Linkage Between Process and Measurement ........................................................................ 8
  2.3 DEVELOPMENT AND APPLICATION OF METRICS ..................................................................... 9
  2.4 EFFECTIVELY COMMUNICATING METRICS INFORMATION .................................................. 12
  2.5 MEASUREMENT AND TEAMS ...................................................................................................... 13
  2.6 MEASURING THE RIGHT STUFF .................................................................................................... 14

3 METHODOLOGY FOR SELECTING APPROPRIATE METRICS ......................................................... 15
  3.1 SYSTEMATIC SELECTION OF METRICS ....................................................................................... 15
  3.2 THE GOAL/QUESTION/METRIC APPROACH (GQM) .................................................................... 15
    3.2.1 Assess the Status of the Environment (Step 1) ...................................................................... 16
    3.2.2 Define and Validate Primary Management Goals (Step 2) ................................................... 16
    3.2.3 Check Appropriateness of the Primary Goals (Step 3) ............................................................ 17
    3.2.4 Define Subgoals from Primary Goals (Step 4) ...................................................................... 17
    3.2.5 Check Consistency of the Goals Tree (Step 5) ....................................................................... 19
    3.2.6 Produce Goals-Related Questions (Step 6) ............................................................................ 19
    3.2.7 Select or Identify Metrics (Step 7) .......................................................................................... 20
    3.2.8 Select or Identify Related Measures (Step 8) ........................................................................ 21
    3.2.9 Develop Measurement Plan for Implementation (Step 9) ..................................................... 22
    3.2.10 Collect and Verify the Data (Step 10) ................................................................................... 23
    3.2.11 Take Appropriate Action(s) (Step 11) .................................................................................. 25
    3.2.12 Feedback Strategy to the Collection Process (Step 12) ....................................................... 25
  3.3 EXAMPLE SYSTEM DESIGN QUESTIONS AND METRICS .......................................................... 25
  3.4 METRIC TEMPLATES OR PROFILES ............................................................................................ 26

4 METRICS FRAMEWORK .................................................................................................................... 32
  4.1 SYSTEMS ENGINEERING MEASUREMENT OVERVIEW ............................................................ 33
4.1.1 Purpose .......................................................................................................................... 33
4.1.2 Systems Engineering Metrics Tree .................................................................................. 33
4.1.3 Example Metrics for Systems Engineering ....................................................................... 39
4.2 TEST ENGINEERING MEASUREMENT OVERVIEW .......................................................... 42
4.2.1 Purpose ................................................................................................................................ 42
4.2.2 Test Engineering Metrics Tree .......................................................................................... 43
4.2.3 List of Core Metrics for Test Engineering .......................................................................... 47
4.3 SOFTWARE ENGINEERING MEASUREMENT ...................................................................... 49
4.3.1 Purpose .................................................................................................................................. 49
4.3.2 Software Engineering Metrics Tree .................................................................................... 49
4.3.3 Example Metrics for Software Engineering ........................................................................ 54
4.4 HARDWARE ENGINEERING MEASUREMENT OVERVIEW .................................................... 56
4.4.1 Purpose .................................................................................................................................. 56
4.4.2 Example Metrics for Hardware Engineering ....................................................................... 65
4.5 MANUFACTURING ENGINEERING MEASUREMENT OVERVIEW ....................................... 67
4.5.1 Purpose .................................................................................................................................. 67
4.5.2 Manufacturing Metrics Tree ............................................................................................... 68
4.5.3 Example Metrics for Manufacturing .................................................................................... 68
4.6 TPM FOR IPTS .......................................................................................................................... 70
5 ANALYSIS TECHNIQUES .......................................................................................................... 73
5.1 CHARTS, GRAPHS, AND TECHNIQUES ............................................................................... 73
5.1.1 Run Charts ............................................................................................................................ 73
5.1.2 Control Charts ...................................................................................................................... 73
5.1.3 Flow Charts .......................................................................................................................... 74
5.1.4 Cause and Effect (Fishbone) Diagrams ............................................................................... 75
5.1.5 Histograms ............................................................................................................................ 76
5.1.6 Pareto Diagrams ................................................................................................................... 77
5.1.7 Scatter Diagrams .................................................................................................................. 78
5.1.8 Check Sheets ......................................................................................................................... 79
5.1.9 X-t Plots ............................................................................................................................... 79
5.1.10 Network Diagrams .............................................................................................................. 80
5.1.11 Gantt Charts ....................................................................................................................... 82
5.1.12 Line Graphs ......................................................................................................................... 83
5.1.13 Pie Charts ............................................................................................................................ 83
5.2 SPREADSHEETS ..................................................................................................................... 84
5.3 DATABASES ............................................................................................................................. 84
6 DEVELOPING METRICS: AN AEROSPACE EXAMPLE ............................................................... 85
6.1 GENERAL ..................................................................................................................................... 85
6.2 THE METRIC PACKAGE .......................................................................................................... 85
6.3 STEPS TO METRIC DEVELOPMENT ...................................................................................... 86
7 METRICS TAILORING AND MANAGEMENT PRIORITIES .................................................... 93
7.1 EXECUTIVE CONSIDERATIONS ............................................................................................ 94
# ILLUSTRATIONS

1.2-1  Metrics Guidebook Context Overview .................................................................2
2.2-1  Weight Weight Technical Performance Measure (TPM)......................................7
2.2.3.1-1  Control Chart (P-Bar) Indicates Performance Trend Problem ..................9
2.3-1  Integration of Goals/Questions/Metrics/Measures .............................................11
2.3-2  Evolution of Customer Needs to Products of Metrics ........................................12
3.2.4-1  Break Primary Goals in Manageable Sub-Goals .............................................17
3.2.4-2  Starting a Goals Template ..............................................................................18
3.2.6-1  Goals Tree ....................................................................................................20
3.2.7-1  Questions Help Identify the Metric(s) ............................................................21
3.2.8-1  Related Measures are Added to the Goals Template ....................................22
3.2.10-1  Metrics Collection Paradigm .......................................................................24
4.1.2-1  Systems Engineering Metrics Tree ..................................................................33
4.1.4-1  Cumulative Budgeted Cost vs. Actual Cost ..................................................39
4.1.4-2  Cumulative Work Scheduled vs. Work Performed .........................................40
4.1.4-3  Cost Estimate at Completion .........................................................................41
4.2.2-1  Test Metrics Tree ..........................................................................................42
4.2.4-1  Staffing Profiles ............................................................................................47
4.3.2-1  Software Metrics Tree ...................................................................................49
4.3.4-1  Software Product Size ...................................................................................54
4.4.1-1  Hardware Metrics Tree ...................................................................................56
4.4.2-1  Cumulative HR/CRs .....................................................................................65
4.4.2-2  Category of HP/CRs .....................................................................................65
4.4.2-3  Time Span of Open HP/CRs ........................................................................66
4.5.2-1  Manufacturing Metrics Tree ..........................................................................67
4.5.3-1  Manufacturing Parts Projected, Ordered and Received ....................................68
4.6-1  TPM Profile ......................................................................................................70
5.1.1-1  Run Chart Showing Shift in Average Errors per Inspection ...........................72
5.1.2-1  Control Chart Showing Process in Control, but Specification Limits Are Not Met........................................................................................................73
5.1.3-1  Flow Chart Showing Normal Sequence of Steps to Turn on Radio .................74
5.1.4-1  Cause and Effect Diagram Showing Relationships Between Root Causes and Effect ........................................................................................................75
5.1.5-1  Pareto Diagram Showing Comparison of Variance in Hours Worked .............76
5.1.5-2  Pareto Diagram Showing Comparison of Task Durations to Optimum Range .................................................................76
5.1.6-1  Scatter Diagram Showing Number of Errors Related to Excessive Overtime ........77
5.1.7-1  X-t Plot Showing Achievement Results for Milestones Over Seven Year Period ...........................................................................................................79
5.1.10-1  Network Diagram Showing Critical vs. Non-Critical Paths and Tasks ...........80
5.1.10-2  Tabular Depiction of Progress Data Including Percentage Completion of Tasks Collected from Network ........................................................................80
5.1.10-3  Graphic Depiction of Milestone Performance Over Time Collected from IPT Meetings .................................................................................................81
5.1.10-4  P-Bar Control Charts Used to Detect Performance Outside Control Limits ....81
1 INTRODUCTION

Measurement is an organization’s fundamental gauge of performance. In this decade, engineering organizations are being bombarded with dozens of “how to” theories for developing successful, quality products. While the experts may differ in the method for building those products, they are in nearly universal agreement that measurement is required to:

1. increase customer satisfaction,
2. align product development and support with the organization’s strategic goals,
3. empower teams to detect process and product problems whose resolution will yield increased customer and organizational satisfaction, and
4. improve processes and products to become more competitive in the marketplace

Measurement, both the art and the science, is the tool by which the performing organization will know whether the product (or system) is meeting its goals. Metrics then, are a composite of relevant quantifiable attributes that communicate important information about the quality of our products, the way we do business, and our productivity. A metrics program provides insight and awareness of opportunities to improve products, performance, processes, and, ultimately, customer satisfaction.

1.1 Guidebook Purpose

The purpose of this *Metrics Guidebook for Integrated Systems and Product Development* is threefold: (1) Capture the experience represented in the National Council of Systems Engineering (INCOSE) Metrics Working Groups, both local and national, where successful measurement programs have been implemented. (2) Support groups establishing new metrics programs. (3) Open a dialog within the Engineering community on relevant measurement in a Systems or Integrated Product Development (IPD) environment.

1.2 Guidebook Background

The guidebook has leveraged on significant activities and research conducted by author member companies, academia, INCOSE and other industry consortiums, literature, and conferences. An overview of the guidebook development process is illustrated in Figure 1.2-1. Thousands of metrics were collected, categorized, and assessed as candidates for the guidebook.

The authors, representing multiple engineering disciplines, further refined these large sets of metrics. The initial material used as a basis for determining the example set metrics (Chapter 4) was a master set of metrics developed by member company projects over a period of five years, tailored by product area and

---


Throughout this guidebook, the definition of “customer” recognizes customers as users of the product or system, either internal or external to the organization.
cooperative with other measurement process initiatives such as Cost of Quality, ISO 9000, Capability Maturity Models for Software and Systems Engineering, and Concurrent
Engineering initiatives. More than three thousand measures were collected in the process of writing this guidebook and developing the example set. The team used techniques such as brainstorming, nominal group techniques, and team consensus to develop a basic metrics set representing the consensus of best measurement content. Organizations engaged in executing or developing a metrics program are encouraged to use this set as an example set, a starting point for developing metrics tailored to be relevant to the products and processes of the organization. The guidebook will be periodically updated to reflect experiences and lessons learned. Topics planned for the second volume are shown in Appendix A.

1.3 Guidebook Objectives

This guidebook addresses benefits, techniques, tailoring, and application of metrics and measurement. The objectives of this guidebook are to provide fundamental measurement constructs, based on understanding system performance and performance in development of systems. This guidebook is intended to provide the following:

1. Understanding of metrics and measurement
2. Description of the measurement process
3. Guidance for using the example metrics set
4. Guidance for tailoring the metrics to project or customer requirements
5. Guidance for defining new metrics through a step-by-step process related to goals and objectives
6. Guidance on how to build a metrics capability (from theory to application)

1.4 Audience

This guidebook is intended for the use of any individual or group responsible for implementing a metrics capability within a commercial, government, or aerospace organization. The authors of this guidebook represent organizations working in these different environments and recognize significant differences in the terminology used to describe the engineering of systems and products. However, the metrics constructs described in this guidebook are useful (and largely common) among the environments, despite differences in terminology.

1.5 How to Use this Guidebook

It is suggested that each reader (especially those individuals new to metrics) take the time to thoroughly read this guidebook in its entirety before launching into any type of metrics implementation.
### 1.6 Guidebook Organization

*Chapter*

1. Introduction: Background, purpose, use, and organization of the guidebook
2. Measurement Process Overview: Answers basic questions about metrics/measurements, basic measurement steps, and what makes a good metric
3. Methodology for Selecting Appropriate Metrics
4. Metrics Framework: Addresses classes of metrics and “example” metrics and measures
5. Analysis Techniques: Defines charts, graphs, and techniques for effectively conveying information
6. Developing Metrics: Provides guidance on how to implement a measurement program that meets the customer’s needs.
7. Tailoring the Metrics Set
8. Managing by the Data: How to Use Metrics
9. Metrics Infrastructure
   Acronyms and Glossary
   References

Appendix A: Guidebook Futures:
   Lists future sections to be included in Volume II of this guidebook. Volume II will be focused on INCOSE cross-functional working group metrics activities.
2 MEASUREMENT PROCESS OVERVIEW

2.1 Understanding Metrics vs. Measures

*Metric* is a composite of meaningful, quantifiable, product or process attributes (measures) taken over time that communicate important information about quality, processes, technology, products and/or resources. However, for a metric to have value, it must have a purpose or a reason to exist – it must yield systematic insight, whether by itself or when combined with other measures. We are then able to take action (to fix a problem, to improve a product or a process), based on the insight that the measurement data is providing. Metrics selection and implementation is specifically designed to improve processes and products.

*Measure only those things from which action can be taken.*

2.1.1 The Three Great Metrics Controversies: Taxonomy, Systems Engineering (SE) Definition, and Integrated Product Development (IPD)

The first metrics controversy recognizes that metrics are categorized using many different taxonomies. The subject of categorization of metrics often becomes so controversial that the importance of measurement is superseded by the language of the taxonomy. For the purpose of this guidebook the authors have chosen a simple division in order to focus the reader on metrics. In this guidebook, three types of metrics are discussed:

1. Technical Performance Measures (TPMs)
2. Planning and Control Metrics

The selection of metrics in these three areas leads directly to the second difficult metrics issue. A common theme throughout industry is the need for a definition of Systems Engineering, from which the metrics of systems engineering might then be easily extracted. This guidebook follows EIA Standard IS-632, *Systems Engineering*, in the belief that “Systems Engineering involves design and management of a total system which includes hardware and software, as well as other system elements. All system elements should be considered in analyses, trade-offs, and engineering methodology.” The same system elements should be considered in design of the metrics set. The guidebook has established two levels of engineering metrics:

1. Systems engineering metrics “owned” by the system engineering function/team member(s)
2. Metrics about which the Systems Engineering function has “need-to-know” in order to produce the product the customer needs. These metrics may be “owned” by another function/team, but their status is critical to an overall understanding of the system performance or system development.

The third significant issue is the impact on systems engineering metrics given an integrated product development (IPD) paradigm. The same issues that are challenging the performing organization from a functional versus product structure are also measurement challenges. The question, “What would you measure differently in an IPD environment” was pivotal to the authors of this guidebook, because each of

---


our organizations is moving to an IPD structure. Interestingly, the information in our existing systems engineering metrics sets remains critical in an IPD environment. There are, however, changes in “ownership” of the metrics, resulting in the levels or tiers described above. Future volumes of the guidebook will add team metrics to the example set.

2.1.2 Attributes of a Good Metric

It is important that a metric and its measures reflect the defined goals and objectives of the organization. How can we tell if a metric is good? A good metric promotes understanding of our performance or progress, as well as our processes, and always motivates action to improve upon the way we do business. This perspective applies from the smallest task through product development to total company operations. A strong metrics program creates an environment in which teams “can make decisions based on data rather than hunches, to look for root causes of problems rather than react to superficial symptoms, to seek permanent solutions rather than rely on quick fixes.”

The following are the basic characteristics of a good metric:

1. It is accepted as having value to the customer or as an attribute essential to customer satisfaction with the product.
2. It tells how well organizational goals and objectives are being met through processes and tasks.
3. It is simple, understandable, logical and repeatable.
4. It shows a trend, more than a snapshot or a one-time status point.
5. It is unambiguously defined.
6. Its data is economical to collect.
7. The collection, analysis, and reporting of the information is timely, permitting rapid response to problems.
8. The metric provides product and/or process insight and drives the appropriate action(s).

In summary, for a metric to be effective it must (1) present data that is useful, thus motivating action(s) to be taken, (2) be able to show status over a period of time, (3) support corporate and product goals and objectives (built from strategic and tactical business plans), and (4) be meaningful to the customer.

2.1.3 What a Metric is Not

1. Metrics are not charts or any other form of display tool, although charts and graphics may display the results of a metric.
2. Metrics are not a team or personnel control tool. Metrics are a process and product control tool. If used against team members, fear, short-term reaction, and “gaming” the system become the output.
3. Metrics are not one-time snapshots or statusing measures. For metrics to be effective, they need to be collected and assessed over time.
4. Metrics are not forever. Different phases of the product lifecycle require different metrics. The team has a responsibility to update the metrics set consistent with the critical processes associated with the product lifecycle phase.
5. Metrics are not schedules, though some schedules lead to good metrics.

---

6. Metrics are not “counts of activity,” although counts of activity or statistics may be significant. Data becomes a useful metric when it is transformed to information or knowledge that can result in action.

2.2 Benefits of Using Metrics

Metrics ensure that detailed measures of engineering processes, performance and product quality are defined, collected, and analyzed to provide quantitative understanding and control in support of improving performance, products and processes. With quantitative measurements, problems and results become more apparent to management, and required actions are clearer to engineers. Metrics help to identify risks early enough in the process to enable correction of the situation or problem before it is out of control – before it affects schedule and cost.

The effectiveness of measurement to identify problems and the use of results to take action for improvement are illustrated by the example in Figure 2.2-1, demonstrating the use of a technical performance measure (TPM) as a risk indicator. Here, the not-to-exceed weight required by the customer's existing configuration is 1600 pounds. Measurements have been taken during the proposal, design and prototyping phases to determine the total weight, using existing equipment, models, and prototypes. The TPM shows the current probable weight relative to the requirement. The original design budget of 1500 pounds had a 7% weight reserve. The last measure shows less than 2% reserve using the revised 1600 pound budget. Using this TPM, graphically showing weights from multiple products being developed by multiple teams, the entire program has visibility into a pending problem and can take action before the design is finalized.7

---

7In this guidebook, the legends associated with graphics, such as Figure 2.2-1, have been eliminated, both to focus the reader on the metric (rather than the technology) and to conserve space. In practice, metrics charts should have a legend.

---
2.2.1 Benefits of Metrics to the Customer

When the customer is provided insight into products, processes, progress, and performance, it raises the customer’s confidence level in the product supplier (or company, or performing organization). The customer wants to know the reality of the situation – how his money is being spent. Metrics not only provide the mechanism that allows the customer to see the whole picture, but the resulting report (information) gives the customer an understanding of technical, cost, and schedule status. It can mean the difference between an aloof or nonresponsive customer relationship versus a teaming atmosphere between the customer and the company. A metric gives the customer insight and allows the customer to become part of the solution.

2.2.2 Benefits of Metrics to the Performing Organization

Metrics help facilitate quality improvement through tracking and monitoring of our current processes and capabilities so that we are able to improve upon them. In a product or project environment, metrics along with a well-defined engineering process provide the infrastructure to promote integration of various team disciplines. Each team member then has visibility into all the functions and areas of responsibility. TPMs provide insight into the technical performance of the product or subsystem. System level (or aggregate) TPMs represent overall performance of the delivered system, often the sum performance of several products. In a similar fashion, planning and control metrics (schedule, resources, cost) at a team level provide insight into accomplishments and problems, while aggregate performance metrics provide knowledge relative to delivery of the product. Process and quality metrics are effective tools for focusing teams on activities that lessen defects and improve cycle time.

Metrics goals are as much about communication as they are about goals and objectives. Interdependent tasks are coordinated with much more ease and accuracy because the team members are more informed. Properly selected metrics support active risk management by early identification of deviations before major cost and schedule impacts occur. A metrics program motivates us to take the required action(s) to stay on track.

2.2.3 Linkage Between Process and Measurement

Metrics are meaningful measures of the performance of a process, product, or activity and form a basis for genuine process improvement. Metrics must be developed by teams with understanding of the process and commitment to take action when required; otherwise even the best metrics are meaningless. The ability to implement change based on accumulated, measured data is the distinguishing factor between taking a measurement and having a useful metric. Metrics must communicate the health of a process and enable the organization or team to distinguish healthy and unhealthy trends for products and processes. All processes exhibit some variation. Continuous improvement of our processes means reduction of variation, within an acceptable set of limits, and that implies measurement.

2.2.3.1 Improvement Programs

The use of metrics is intrinsic to quality/improvement programs including ISO 9001 and the Software and Systems Engineering Capability Maturity Models (CMM). The metrics tell the performing organizations where genuine improvement has occurred. Measurement also provides clarity in understanding the problems so that the organization’s goals can be reached. Using metrics for monitoring and tracking provides increased visibility into the product’s progress and quality. Depicting and understanding process variation with metrics is essential to systematically improving the process. Metrics techniques assist in depicting both the normal and abnormal variations in any process.
Normal variations are usually process related. These types of variations may include work environment, communications, work methods, materials, and reliability of equipment. Internal processes can often be improved to avoid repeating the same problem.

Abnormal variations are nonroutine or unusual causes. For instance, an increase in errors may be the result of a new employee having been given the responsibility of a critical task. In this situation, the process itself may not require a change, but the additional training of personnel may be in order. Metrics help identify the point of insertion in the process where the problem occurs. Abnormal variation may show up during monitoring of a process that shows considerable normal variation as well, as illustrated by Figure 2.2.3.1-1. In this case, a custom circuit card product line that has zero defect/modifications percentages between 72 and 98% over a significant period of time suddenly dips below the program's established lower control limit (LCL). Detailed examination of the process being monitored should be conducted for possible causes of this abnormal variation before the LCL is breached.

![Circuit Card % Board Modifications]

Figure 2.2.3.1-1. Control Chart (P-Bar) Indicates Performance Trend Problem

### 2.3 Development and Application of Metrics

Selecting existing metrics or developing new metrics should be directly related to the product’s or organization’s critical success factors, usually defined as technical, cost, or schedule targets or requirements. Those factors are the activities, attributes, or characteristics of the product(s) delivered to the final customer that, taken together, define the end result achieved and the success or failure of the organization's goals. Figure 2.3-1 illustrates a top-down flow transforming an organization's vision and mission into goals and objectives and, from there, into key performance metrics. The performing organization requires metrics that are relevant to the vision and mission of the organization that yield insight into customer satisfaction. Goals developed from the organization’s mission are the source of metrics that the performing organization will really support. The metrics implementation (collection, analysis, extrapolation, and action) in turn provides insight into process and product improvements, yielding increased customer satisfaction. But how does one apply practical techniques to accomplish this? Two processes that can be used to get this alignment between customer satisfaction and organizational success are Quality Function Deployment (QFD) and
Goal/Question/Metric (GQM). The GQM method is a generalization of the Factor/Criteria/Metric framework developed at Rome Labs. Rome Labs’ approach was to identify general goals (factors) of interest to all projects.

QFD is a methodology and a tool designed to represent the voice of the customer and to translate customer requirements or needs into technical requirements for product development. The strength of QFD can be seen very rapidly because the structure aligns requirements to products, yielding insight into (1) areas where the requirement is satisfied by the product and (2) the holes where either a product or component exists with no reciprocal requirement or the requirement exists without a reciprocal component.

A further strength of QFD is the prioritization of the requirements. Often, our customers have many requirements, not all of which can be met efficiently at the same time. With a prioritized set of requirements, the customer and the performing organization have a mechanism for making sound trades and subsequent development decisions. The prioritized technical requirements frequently imply the key TPMs. These measures represent the most critical product or system performance requirements from the perspective of the customer.

There may be other critical product TPMs that are not represented in the QFD, often because they represent a requirement at a lower level in the product tree or because they represent an essential technology that is not specific to the customer expression of the requirements. Establishing the complete set of TPMs for the product is essential. Figures 2.3-1 and 2.3-2 are designed to show the evolution of a measurement set being driven by the needs of the customer, the goals of the performing organization, and industry standards and practice.

QFD and GQM represent two of many techniques that can be used to develop a metrics set. A third technique, based on operational definition developed by the Air Force, is described in Chapter 6, Developing Metrics: An Aerospace Example. This guidebook uses GQM to develop an example set in order to provide a broader view of metrics that may be of utility to a wide range of performing organizations. GQM is an approach used to identify and measure progress toward critical product, system, or organizational goals. This guidebook expands upon the GQM approach (in Chapter 3, Methodology for Selecting Appropriate Metrics, and Chapter 4, Metrics Framework) by showing example metrics sets based on GQM work done in teams where system, software, hardware, and manufacturing functions were represented in the groups. The intent of those sections is to support performing organizations in the development of their own metrics sets.

---

8ami consortium, Centre for Systems and Software Engineering, South Bank University, ami-application of metrics in industry, a quantitative approach to software management, a progressive guide for managers and engineers involved in software development, PAC Graphics Limited, London, United Kingdom.

Figure 2.3-1. Integration of Goals/Questions/Metrics/Measures
2.4 Effectively Communicating Metrics Information

"No matter how well you have prepared what you are going to say or how skilled you may be in your speaking delivery, your audience still has the capacity to daydream: they can think faster than you can speak...use graphic visuals whenever possible to show relationships."\textsuperscript{10}

— Mary Munter

Metrics collected and not communicated have little value. Most organizations have significant amounts of data that fall in this category (whether or not they are called metrics). The best measures of the utility of a metric are its pertinence to the population from which the data was collected and the use to which the information is put. Annual, mandated sweeps of information collection are unlikely to improve the product or process from which the data was extracted. Useful information by definition must improve a product or a process in a timely way.

Information (not just data) from which knowledge can be extracted and action taken must be presented in a manner that clarifies and fairly presents findings. The delivery of the information can be verbal, written, or electronically distributed. However, the needs remain the same:

1. Information that is useful
2. Information that is objective, accurate, and internally consistent

3. Information that is regularly available to all members of the team

Communicating metrics information can be accomplished in several ways. Dissemination of the information can be by briefing, bulletin board, on-line access, or even newsletter. The key is to disseminate useful information as rapidly as possible to the population that can benefit (take action) from the information.

2.5 Measurement and Teams

"...initiate open-ended discussions about the performance and purpose that can turn you into a team. Reexamine the goals of the group: are they clear, specific, measurable, and performance-focused?"

— Katzenbach and Smith

The focus of this guidebook is on the linkage between the organization's goals and the establishment of meaningful and useful metrics to create knowledge of how well or poorly those goals are being met. Hammer and Champy take this concept one step further by linking values and beliefs of the team members to effect process reengineering to meet the organization's goals. In Table 2-1 Hammer and Champy’s reengineering value beliefs are shown in the left column, and the right column highlights the associated measurement construct.

<table>
<thead>
<tr>
<th>Reengineering the Corporation, Hammer and Champy</th>
<th>Measurement Constructs</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Customers pay all our salaries: I must do what it takes to please them.”</td>
<td>Establish the metrics of the organization, product or program directly in line with customer(s) real measures of value.</td>
</tr>
<tr>
<td>“Every job in this company is essential and important: I do make a difference.”</td>
<td>The members of the product development team are responsible for both establishing the appropriate measures and for ensuring that the team delivers measurable value to the customer.</td>
</tr>
<tr>
<td>“Showing up is no accomplishment: I get paid for the value I create.”</td>
<td>Metrics that do not provide this insight must be reworked until they do reveal the vital signs of the product or process.</td>
</tr>
<tr>
<td>“The buck stops here: I must accept ownership of problems and get them solved.”</td>
<td>It is the team's responsibility to use metrics to determine, measure and correct problems. Sending the metrics report to management is not solving the problem.</td>
</tr>
<tr>
<td>“I belong to a team: We fail or we succeed together.”</td>
<td>Metrics are not a management control tool; they are a product and process improvement tool. Metrics used as a personnel control tool provoke &quot;short-term thinking, misguided focus, internal conflict, greater fear, and blindness to customer concerns.&quot;</td>
</tr>
</tbody>
</table>

This guidebook is structured around the measurement constructs shown in Table 2-1 following the process shown graphically in Figure 2.3-2. An effective metrics program is structured around customers’

---

needs and the organizational mission and vision and is executed as part of the organization's standard processes.

### 2.6 Measuring the Right Stuff

“The increasing complexity of today’s world leads many managers to assume that they lack information needed to act effectively. I would suggest that the fundamental “information problem” faced by managers is not too little information but too much information. What we most need are ways to know what is important and what is not important, what variables to focus on and which to pay less attention to – and we need ways to do this which can help groups or teams develop shared understanding.”

— Peter M. Senge, *The Fifth Discipline*[^14]

A strong metrics program allows the organization or project to measure progress toward any goal, even those unique to the company, organization, project, or task. This guidebook expands and builds upon the GQM approach in Chapters 3 and 4, showing an example metrics set tailored to meet the needs of a performing organization, with linkage from goals, objectives, and decision making to measurable entities to include resources, products, and processes of the program or organization.

---

3 METHODOLOGY FOR SELECTING APPROPRIATE METRICS

There are many metrics. Selecting ones that provide insight into an engineering process and product development is not an easy task. The Goal-Question-Metric (GQM) approach is a top-down, information-need-based paradigm that has been used extensively to define metrics for software development and process improvement. This GQM approach shown below was developed by the ami consortium as a means for selecting metrics. Additional material presented here is based on work performed at the Software Productivity Consortium.

3.1 Systematic Selection of Metrics

Metrics should be selected using a rational, top-down process, which starts by identifying an information need of a specific class of user. The GQM is a method (described in section 3.2) for the systematic selection of metrics. It has been applied widely and successfully for process improvement and software metrics programs. The top-level goals used in the GQM paradigm are often ones associated with project control and process improvement. The GQM paradigm helps define metrics appropriate to a stated information need. It requires definitions of who needs to know what information, and why and when they need to know it. Any metric selected should be meaningful to its users, repeatable, and accurate in terms of its representing the attribute of the product or process it is supposed to represent. The GQM paradigm includes the following steps:

1. State the information goal: Identify the information consumer groups (stakeholders) and determine what each wants to know and wants to do with the information.
2. Ask the question: What question(s) should be asked to determine whether the goal is being met?
3. Identify the specific parameters that must be measured to answer the question posed in step 2: What metric is needed (directly or indirectly) and what must be measured to obtain it?
4. Apply the metrics selected, evaluate their usefulness, and go to step 1 or step 2, if indicated.

It is important to recognize that selecting metrics is typically an iterative process. Each cycle ends with an evaluation of the measurement system that has been put into place.

One should differentiate measurement goals from organizational goals. Measurement goals are derived from organizational goals. For example, if a goal of the project is to produce a reliable system, then a measurement goal is to determine (actually estimate with some stated degree of confidence) the system's reliability.

3.2 The Goal/Question/Metric Approach (GQM)

The process of developing and applying a metrics program consists of several steps or activities:

1. Assess the Status of the Environment

---

15 The 12 steps described here were called the Route Map, when developed by the ami consortium, Centre for Systems and Software Engineering, South Bank University, ami-application of metrics in industry, a quantitative approach to software management, a progressive guide for managers and engineers involved in software development, PAC Graphics Limited, London, United Kingdom.

2. Define and Validate Primary Management Goals
3. Check Appropriateness of Primary Goals
4. Define Subgoals from Primary Goals
5. Check Consistency of the Goals Tree
6. Produce Goals-Related Questions
7. Select or Identify Metrics
8. Select or Identify Related Measures
9. Develop Measurement Plan for Implementation
10. Collect and Verify the Data
11. Take Appropriate Action(s)
12. Feedback Strategy to the Collection Process
A general outline of these activities is shown below and expanded in subsequent paragraphs.

3.2.1 Assess the Status of the Environment (Step 1)

In order to improve upon processes and products, the current status of the environment must be assessed. Assessments are usually conducted by a small team (5 to 10 individuals). Assessment is any general activity that collects, examines, and sorts information against a scale. For example, a scale can show degrees of risk, costing ratios, or maturity levels of management practices.

Assessment also involves an examination of the environment from one or more particular viewpoints: process, product, and communications. Process will take into consideration engineering processes, scheduling practices, roles and responsibilities, and so on. Product assessment will include such characteristics as quality criteria and is generally customer oriented. Communications is the information flow within and between teams on a program or project. Once we come to an understanding of our present position, then we are better able to determine our target position and the areas of improvement.

Tools for assessing the engineering environment include ISO, SEI, or Baldridge criteria to provide guidance in determining areas of improvement within engineering disciplines. Also starting to emerge are online tools that can assist companies in analyzing their business health and status.

Assessing the environment is particularly important at the corporate and organizational level of the metrics paradigm. Once we understand where we are as a performing organization, we can then identify our primary goals and objectives for the areas of improvement at the program and project levels.

3.2.2 Define and Validate Primary Management Goals (Step 2)

Corporate goals and objectives, as well as the assessment results, provide input and direction for defining the primary goals of the organization. In turn, organizational goals provide direction at the program and project levels. For the sake of simplicity, the following paragraphs will address the remaining steps, for the most part, within the context of a product environment. However, these same steps can be applied at any level (corporate, organization, program, project, team, or even task).

Define primary management goals – There are two classes of goals: Control (Resource) and Change (Achievement). Control goals are quantitatively measured using verbs such as evaluate, predict, or monitor. The verb evaluate addresses the past, monitor the present, and predict the future. Change goals are quantitatively measured using verbs such as increase, reduce, achieve or maintain.
Primary goals should be defined by management and the assessment team. An obvious goal or metric is not always the most suitable for solving your problems. Goals and metrics must be applied within the context of the environment. At the program level, the primary goals should be established based on organizational goals, assessment results, and customer’s direction. It is always important to consider what is useful to your particular program, and to tailor organizational goals accordingly.

An important step after identifying primary goals is to translate them into benefits for the participants or the customer. Understanding how the goals affect the participants and the customer is as critical as defining the goals themselves. Example questions to ask during the translation are, “Who is the customer?” “What are the customer’s expectations?”, and “What results will the customer be interested in?” Participants are those who will be affected by the goals and are responsible in some way for implementing the actions to meet those goals.

3.2.3 Check Appropriateness of the Primary Goals (Step 3)

Are the goals consistent with the assessment findings? Are your program goals in line with your organizational goals and the customer’s goals? Are they clear, precise, quantifiable, simple and understandable? Are there sufficient resources and commitment to reach these goals within the defined time? Or should a specific goal(s) be considered for future programs when there might be more resources available?

3.2.4 Define Subgoals from Primary Goals (Step 4)

Each primary goal is analyzed and broken down into manageable subgoals, forming a goals tree as illustrated in Figure 3.2.4-1. At the end of each tree branch a table of questions will be developed addressing how this particular goal will be met.

![Figure 3.2.4-1. Break Primary Goals in Manageable Subgoals](image)

An important element in this analysis is to set goals that correspond to areas of responsibility and activities of the teams or individuals as well as customer quality requirements and processes that affect the quality. Using a goals template, list each subgoal on a separate template as shown in Figure 3.2.4-2, leaving room for the related questions, selected metrics, and measures.
Primary goals are broken down into subgoals through a step-by-step analysis process usually accomplished through a series of participant working groups conducted by the team lead.

Primary goals are analyzed according to products, processes, or resources (these are referred to as entities), and managed by the individuals active in the decision making process and those who will use the measurement data. Each participant during decision making will take into consideration several factors to include products under their responsibility and the characteristics of those products, processes within the organization, and resources they are managing. These considerations help identify the subgoals and effectively represent the viewpoints of the various participants.

For each primary goal, it is a good idea to create a table to list the various products, processes or resources represented by each participant. Take for example, the primary goal to Monitor and Improve Productivity While Maintaining Quality. Many participants will be involved in decision making and managing processes, products, and resources to accomplish this goal (Chief System Engineer (CSE), Quality Assurance, Product Manager, etc.). For this example, we will take the viewpoint of the Chief System Engineer. What are the entities managed by the CSE? What processes, products, and resources will the CSE be responsible for to meet this primary goal? Table 3-1 provides a brief example for the Process of Inspections for Specifications. One of the many questions that should be asked for Inspections is given in the example.
Table 3-1. Entity Table Example

<table>
<thead>
<tr>
<th>Entities Managed by Chief System Engineer</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESSES</td>
<td></td>
</tr>
<tr>
<td>Specification Inspections [...]</td>
<td></td>
</tr>
<tr>
<td>PRODUCTS</td>
<td></td>
</tr>
<tr>
<td>Specifications, ICD, IFS [...]</td>
<td>Are the product specifications inspected periodically throughout the project’s lifecycle to detect problems early on that might affect system technical performance later?</td>
</tr>
<tr>
<td>RESOURCES [...]</td>
<td>...</td>
</tr>
</tbody>
</table>

Taking the Process of Inspections, what subgoal is derived consistent with the parent primary goal? The answer for this example is *Detect Impediments to System Technical Performance*. Add this subgoal to the goals tree. This process should be iteratively repeated for each participant’s entities (processes, products, and resources).

3.2.5 Check Consistency of the Goals Tree (Step 5)

When the participants in the working group reach consensus that all viewpoints have been appropriately addressed, review the consistency of the goals. Is there balance between and within the branches? Is the granularity similar within each branch? Does the breakdown effectively represent the viewpoints of each of the participants? Is there a clearly defined communications flow? If the answer to any of these questions is no, readdress the area in question through iterative refinement until the participants are in consensus on the results.

In addition, ensure that each goal faithfully represents its parent goal on the tree and that both belong in the same class.

3.2.6 Produce Goals-Related Questions (Step 6)

Taking the table produced for entities (example, Table 3-1), expand and review the related questions. Add these questions to the goals tree as shown in Figure 3.2.6-1 for each subgoal.

Team consensus is important. Buy-in from all levels of the project helps eliminate careless cooperation and possibilities of skewed data inputs.

Two groups from the same project can arrive at different sets of questions. Limit the team to no more than 10 individuals, otherwise, the goal/question/metric cycle will become a frustrating and non-productive experience.

When questions have been identified that can’t be answered unless there are organizational changes, then set aside that particular question until it becomes a relevant part of the organization, otherwise the metric associated with that question contains no value to the improvement of the current status of the company, organization, program, or project.
3.2.7 Select or Identify Metrics (Step 7)

Limit the metrics to those that correspond directly to significant goals, thus minimizing collection and analysis costs and making the collection process much easier. Clear definition of goals increases acceptance of measurement by program leads and engineers alike. Measurement for the sake of measurement is demotivating and not readily accepted.

There will be two types of metrics identified by this step: subjective and objective. Subjective metrics usually apply to quality; they have no absolute values. The collection of subjective metrics is usually accomplished through user questionnaires, interviews, or surveillance monitoring of the user. An example of a subjective metric is usability. Objective metrics, on the other hand, are easily quantifiable and have definite values. It is recommended that objective metrics be used whenever possible.

Use well-tried metrics, especially when first invoking a metrics program. A good example of this is labor hours. Labor hours are often collected by finance organizations. Exercise the new metric in tandem with a known baseline, using supporting data already collected during the normal process of conducting business. For example, if defects are the primary metrics focus, use a denominator of labor hours. In this way, the significance of the new metric is understood in a known context.

Using the verified goals tree, derive the metrics to match the subgoals on the lowest levels of the tree, using the related questions as a specification for the metric. As illustrated in Figure 3.2.7-1, the questions are used to identify metrics selected from the Example Metrics Set or to develop new metrics. The metrics that have been derived from the lowest level of the goals tree are usually combined with other metrics and aggregated upward to provide quantitative results associated with a higher level goal. A good example is the cost of a subsystem. Combining the cost of all subsystems will provide information on the total cost of the project.
### Subgoal 1: Detect Impediments to Productivity

**Questions:**
- **Q1:** Are products and processes inspected throughout the project's lifecycle to detect problems early that might affect productivity later?
- **Q2:** How many errors are found (over time)?
- **Q3:** How effective are inspections in finding critical as well as minor problems early in the project?

**Metric:**
- **M1:** Inspection - Defect Profiles
- **M2:** Inspection - Effectiveness

**Measures:**
- **Ms1:**  
- **Ms2:**  

---

The Example Metrics Set was identified through a series of working groups with participants from each discipline. The Example SE Metrics Set relates directly to systems engineering with reference ("need-to-know") to metrics from the areas of software engineering, test engineering, hardware engineering, and manufacturing. Additional metrics sets will be identified and added during revisions to this guidebook. The Example Metrics Set was developed using the attributes for good metrics described in section 2.1.2 as the baseline from which to start a metrics program, tailoring it to meet customer and program requirements.

The Example Metrics Set is a subset of, and is derived from, a master list of metrics known as the Metrics Superset. The Metrics Superset has been developed over a period of years by member companies adopting perspectives from industry, academia, and national organizations.

An example is provided for using and tailoring the Example Metrics Set in Chapter 4.

**Notes to Remember** – Generate new metrics if existing metrics do not relate well to a specific goal(s) of the program. Modify existing metrics if they do not adequately reflect the intent. When creating new metrics, rate them against the previously defined "attributes for a good metric” discussed in section 2.1.2.

### 3.2.8 Select or Identify Related Measures (Step 8)

Identify measures that make up each metric. Again, using the questions as a specification for the metric, answer each of the questions through quantitative or qualitative measures. Quantifiable measures are added to the goals template in Figure 3.2.8-1. The metrics and measures shown are selections from the Example Metrics Set. Measure 1 (Actual No. of Major Defects per Defect Type (per Phase)) is related to Metric 1 (Inspection – Defect Profiles) and answers Question 1. Measure 2 (Actual No. of Major Defects (per Phase)) is related to Metric 2 (Inspection – Effectiveness).
Inspection) is derived from Metric 2 (Inspection – Effectiveness) and answers the first half of Question 3. Adding a measure called “Actual No. of Minor Defects (per Inspection)” will answer the last half of Question 3. The process of measurement definition continues in this manner.

**Subgoal 1: Detect Impediments to Productivity**

**Questions:**

**Q1:** Are products and processes inspected throughout the project’s lifecycle to detect problems early that might affect productivity later?

**Q2:** How many errors are found (over time)?

**Q3:** How effective are inspections in finding critical as well as minor problems early in the project?

**Metric:**

- M1: Inspection - Defect Profiles
- M2: Inspection - Effectiveness

**Measures:**

- Ms1: Actual No. of Major Defects per Defect Type (per Phase)
- Ms2: Actual No. of Major Defects (per Inspection)

---

**3.2.9 Develop Measurement Plan for Implementation (Step 9)**

The measurement plan is an important basic step for implementing a metrics collection process on a program. The plan may be a stand-alone document or incorporated in the Product Development Plan, Systems Engineering Management Plan (SEMP), Hardware Development Plan (HDP), or Software Development Plan (SDP).

Previous steps (1 through 8) should be used as the starting point for writing the program or project measurement plan. Much of the information provided in this guidebook can be tailored and also used in the program plan. A good example of what a measurement plan should contain is structured as follows:

**Part 1: Objectives of the Measurement Plan**

In this part, the context, assessment conclusions (when required), goals and questions are detailed. A summary of the results of all the work carried out in the previous steps are included. This information is important for correct metrics definition and exploitation of measurement data.

**Part 2: Metrics Definitions**

---

Suggested by the ami consortium, “ami application of metrics in industry.”
A precise definition and analysis procedure is given for each metric. A metric is specified by a question and is linked to one or more goals. Those who analyze the metric and who received the results are identified. How and when the measurement data is to be presented are stated. Each metric is made up of one or more collected or computed primitive data items, so each primitive data item now has to be clearly identified with a precise collection procedure.

**Part 3: Responsibilities and Timescales**
All major responsibilities and milestones and collection points should appear in this section. In some cases those responsible for different activities will be able to draw up their own plans.

**Part 4: References**
References and support material are identified.

**Part 5: Logbook for Measurement Activities**
A logbook kept during the project helps in the monitoring activity and in the analysis of the metrics.

Validate the measurement plan by putting it into action and implementing the measurement process. It is important that all levels of management and their staff support implementation of the measurement plan – it is a crucial factor for success. A regular metrics collection and reporting schedule should be established, requiring cooperation of the development team(s), management, and the metrician/or process group.

### 3.2.10 Collect and Verify the Data (Step 10)

The metrics collection process consists of several steps as illustrated in Figure 3.2.10-1. Raw data is collected and results are calculated, presented in review, and analyzed. Based on the analyzed results, appropriate action is taken to ensure that the program or project goals are met.

*Identify Input Data and Source* – The raw data to be collected is determined by the measurements identified in the previous steps, tied directly to goals and questions. The collection source of raw data is often already established and requires no additional collection procedures. For example, at most of the member companies, labor hours are collected by Finance. Costing information is collected as part of our proposal and life cycle processes, calculated through the use of parametric models; schedule and milestone data are captured in our planning process, and so on. The first step is to identify already established collection points within the company. From there, define new collection procedures for data that is not already collected automatically from an established source.

*Collect Raw Data* – Use data collection forms that are simple to read and complete. Multiple fields of input may deter getting the appropriate data filled in on time. Also, collection forms and tools should be developed or available on common platforms using standard interfaces for data transfer between tools. Give preference to spreadsheet packages, commercially available products, and database packages, especially for trend analysis and target metrics programs. Make the collection process as simple and unintimidating as possible.

*Verify the data* – This is accomplished by checking the accuracy and completeness offered on the data collection forms. Check for any data values that are unusual or repeated patterns that might indicate carelessness.
Spreadsheets and databases support verification of the data as well as graphical representation for conveying results quickly and clearly.

After verification, enter the data into the database or other storage capacity.

*Extrapolate, Analyze, and Review Measurement Data* – Periodic calculation is made on the collected raw data, and the results are presented for analysis. Analysis begins with an effective presentation of the measurement data. Presentation is usually in graphical format and may be distributed for review or, preferably, presented in a team forum.

Graphical display of the collected data should reflect only the data relevant to the group's particular goals. Presented data can show estimates (using past data), error rates (optimally decreasing to zero), progress (optimally increasing to 100%).

An important factor to remember is to conduct the analysis within context of the project – using background information, involving the project team, and comparing the measurement data against actual
goals and assumptions. This approach will serve to validate the collected data and avoid using data void of reason.

3.2.11 Take Appropriate Action(s) (Step 11)

The appropriate action is dependent upon the analysis of the measurement data. By relating the data to goals and assumptions, improvement or corrective action can be taken. Measurement results often initiate additional questions and may prompt the team to derive new goals. Each iteration should provide new insight, causal analysis if possible, and awareness of the overall process and should allow the team to improve upon the process, from a metrics and goals standpoint.

Also, reassess the project’s metrics plan, viewing the costs and benefits of measurement, validating the selected measures (so they directly correspond to the subgoals), and refining the use of the statistics and models.

3.2.12 Feedback Strategy to the Collection Process (Step 12)

The goals, objectives and results of the measurement cycle should be reviewed periodically. These reviews will help improve the measurement plan and increase the effectiveness and efficiency of measurement. In turn, efficiency and motivation will lead to predictable and controlled metrician costs, and part of the cost is transferred from existing sources, keeping these costs at a minimum.

3.3 Example System Design Questions and Metrics

It is important to recognize that systems engineering is a management technology, i.e., systems engineering involves the interaction of engineering science, the development organization, and the application environment (Sage 1992). The interactions among these three elements are in the form of information, and some of this information will be in the form of metrics for the system. The quantification of system characteristics, i.e., system measurement, is a necessary part of system development and the systems engineering process.

The overall system development process and its systems engineering component can be described as a network of activities. This section provides an example of selecting some of the metrics for the system design activity of the systems engineering process using the GQM paradigm.

A statement of requirements for the overall system is the input to the system design activity. The design activity partitions the requirements among hardware, software, and procedures. Outputs could include product design description, software requirements specification, and/or hardware requirements specification.

A top-level or organizational goal might be to manage a development project. A subsidiary (detailed) or measurement goal would be to determine the status of the system design activity. This is a constituent of the larger scope question, “What is the status of the project?” Table 3-2 provides more detailed questions and metrics selected to answer them. The GQM paradigm is used to select the right questions to get the information needed for decision making.
### Table 3-2. Example System Design Questions and Metrics

<table>
<thead>
<tr>
<th>Question</th>
<th>Metric Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. What has been processed by this activity?</td>
<td>Metric Category: Number of system requirements statements evaluated (size)</td>
</tr>
<tr>
<td>2. What has been produced by this activity?</td>
<td>Metric Category: Number of software and hardware requirements statements developed (size)</td>
</tr>
<tr>
<td>3. How much effort has been expended on this activity?</td>
<td>Metric Category: Labor Hours (effort)</td>
</tr>
<tr>
<td>4. How much time has been spent on this activity?</td>
<td>Metric Category: Clock Hours (schedule)</td>
</tr>
<tr>
<td>5. How many times has this activity been repeated?</td>
<td>Metric Category: Number of Repetitions (schedule)</td>
</tr>
<tr>
<td>6. How much rework has been done?</td>
<td>Metric Category: Rework Labor Hours (effort)</td>
</tr>
<tr>
<td>7. How much more has to be done?</td>
<td>Metric Category: Number of system requirements statements to be evaluated (size), Labor Hours to be expended (effort)</td>
</tr>
<tr>
<td>8. When will the activity be completed?</td>
<td>Metric Category: Clock Hours (schedule)</td>
</tr>
</tbody>
</table>

A successful systematic measurement program that supports management and engineering decision making will result in systems that are developed on time, within budget, and do what they are supposed to do.

### 3.4 Metric Templates or Profiles

Metric templates or profiles, such as those shown in the INCOSE Metrics Catalog\(^{18}\), are basic metric descriptors that define the metrics and provide other essential information about the metric. These profiles can be sophisticated or simple. They are tailor able to the specific needs of the project or in accordance with the customer’s requirements. Table 3-3 reflects a robust metric profile format that can be used as is or tailored in accordance with customer and program requirements and goals.

---

\(^{18}\)The INCOSE Metrics Catalog will be included in next version of this guidebook. A preliminary draft version of the Catalog is currently in review by the INCOSE Metrics Working Group.
Table 3-3. Metric Profile Example

PROJECT NAME: _________________________

**METRIC PROFILE**¹⁹

(Note: This metric profile may be used (tailored) for inclusion within a proposal to address the customer’s requirements, or as one of the core metrics to monitor and track project progress, performance, risk and overall success.)

<table>
<thead>
<tr>
<th>METRIC NAME:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE OF METRIC (Indicator):</td>
<td></td>
</tr>
<tr>
<td>METRIC DEFINITION:</td>
<td></td>
</tr>
<tr>
<td>RELATED GOALS:</td>
<td></td>
</tr>
<tr>
<td>RELATED QUESTION(S):</td>
<td></td>
</tr>
<tr>
<td>METRIC/MEASURES:</td>
<td></td>
</tr>
</tbody>
</table>

¹⁹The Metrics Profile format is adapted from a metrics form developed by Tandem Corporation, Sunnyvale, California. This format is an alternative to the Metrics Catalog template. Each performing organization should modify or develop a format that meets the organization’s unique needs.
### USAGE:

<table>
<thead>
<tr>
<th>Table 3-3. Metric Profile Example (Continued)</th>
</tr>
</thead>
</table>

### NEED TO KNOW:

| First Level Benefactor (Metric Owner): |
| Second Level Benefactor: |
| Third Level Benefactor: |
| Fourth Level Benefactor: |

### SOURCE OF RAW DATA:

| Requirement: |
| Manual Estimates/Counts: |
| Financial Reporting System: |
| Project or Division Database: |
| Schedule: |

### COLLECTION POINTS:

| One Time Collection Points: |
| [Pre-Bid, Engineering Bid (Prop), Best and Final Offer (BAFO)] |
### Periodic Collection Points:
Product Concept Phase, Product Design, Development, Integration and Test, Alpha/Beta Phase, Field Support, or Proposal, Systems Requirements, System Design, Preliminary and Critical Design, Test Readiness Review, Functional Configuration Audit Phases [m(monthly), biw(bi-weekly), w(weekly), d(daily), r(review), ms(milestone)]

### Action(s):

<table>
<thead>
<tr>
<th>LEVEL OF EFFORT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(Labor Hours)</strong></td>
</tr>
</tbody>
</table>

---

**INCOSE-TP-1995-002-01 (originally INCOSE TM-01-001)**

---

**INCOSE-TP-1995-002-01 (originally INCOSE TM-01-001)**
### Table 3-3. Metric Profile Example (Continued)

**COLLECTION RESPONSIBILITIES:**

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsible Party</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Management Office (PMO):</td>
<td></td>
</tr>
<tr>
<td>Chief Engineer (CE)</td>
<td></td>
</tr>
<tr>
<td>Metrics Specialist (if needed):</td>
<td></td>
</tr>
<tr>
<td>Project Manager:</td>
<td></td>
</tr>
<tr>
<td>Technical Lead:</td>
<td></td>
</tr>
<tr>
<td>Supervisor:</td>
<td></td>
</tr>
<tr>
<td>Designer:</td>
<td></td>
</tr>
<tr>
<td>Programmer:</td>
<td></td>
</tr>
<tr>
<td>Documenter:</td>
<td></td>
</tr>
<tr>
<td>Reviewer:</td>
<td></td>
</tr>
<tr>
<td>Tester:</td>
<td></td>
</tr>
<tr>
<td>Configuration Management:</td>
<td></td>
</tr>
<tr>
<td>Quality Assurance:</td>
<td></td>
</tr>
</tbody>
</table>

**TOOLS:**

<table>
<thead>
<tr>
<th>Tool Used for Computing Final Metrics (e.g., spreadsheets, statistics tools, and other tools)</th>
<th></th>
</tr>
</thead>
</table>
Table 3-3. Metric Profile Example (Continued)

**REPORTING/PRESENTATION OF RESULTS:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphics/Displays (histograms, pie charts, scatter plots, etc.):</td>
<td></td>
</tr>
<tr>
<td>Interpretation:</td>
<td></td>
</tr>
<tr>
<td>Mode of Communication:</td>
<td></td>
</tr>
<tr>
<td>Engineering Reporting Format:</td>
<td></td>
</tr>
<tr>
<td>Management Reporting Format:</td>
<td></td>
</tr>
<tr>
<td>Executive Management Reporting Format:</td>
<td></td>
</tr>
<tr>
<td>Customer Reporting Format:</td>
<td></td>
</tr>
</tbody>
</table>

**STORAGE OF RESULTS:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Storage:</td>
<td></td>
</tr>
<tr>
<td>(historical or project database, etc.)</td>
<td></td>
</tr>
<tr>
<td>Access Path:</td>
<td></td>
</tr>
</tbody>
</table>
"The fact that a product is six months late and $2 million over budget does not tell anyone what went wrong or what to do next." — Christopher Meyer

4 METRICS FRAMEWORK

This section represents a set of metrics and measures intended to support product development, service, and management organizations in development of their own appropriate metrics set. The subsections of this chapter represent functions commonly represented on product development teams, including:

1. Systems Engineering
2. Test Engineering
3. Software Engineering
4. Hardware Engineering
5. Manufacturing Engineering

The type of metric, relative to the discussion in Chapter 2—engineering process and quality, planning and control, or technical performance measure— is identified with an abbreviation in parentheses following the metric. In the next release of this guidebook, the Metrics Working Group hopes to add additional IPT functions including Logistics, Specialty Engineering, and Material Acquisition and Purchasing and a section on Executive Metrics. The addition of these functions as well as other disciplines is essential to have full visibility into the entire product development and support processes. Additionally, a catalog of metrics examples will be included in an appendix.

This section of the guidebook could have been organized in several different ways. The decision to display the information within functional areas with TPMs, Planning and Control, and Process and Quality Metrics shown in each section was done at the request of our initial reviewers, who found utility in being able to discuss metrics a function at a time to incrementally assemble the IPT’s metrics set. This approach follows the System Engineer "owned" metrics versus "need to know about the system" metrics breakout discussed in Chapter 2. The System Engineering representative on the IPT is often (but not always) the process owner for the metrics shown in Section 4.1. If responsibility for those processes is held elsewhere, then the responsibility for the metric is also held elsewhere. In a like fashion, metrics described in the test, software, hardware and manufacturing sections may be the responsibility of different groups within other organizations. The intention of this section is to provide the reader with ideas from which to structure their own unique metrics set.

Finally, a note on the "silver bullet" metric. In the past five years, thousands of metrics have been discussed, displayed, and disputed in many forums. The authors of this guidebook have been asked repeatedly for the one most important metric. Sometimes, the more merciful inquisitors ask for the top 10 metrics. System engineers have particular affection for requirements volatility, TPMs and cycle time. Today, our best answer is still the first answer.

4.1 Systems Engineering Measurement Overview

This section provides an overview of the example systems engineering metrics, their relationship and support to an organization’s goals, and selected metric examples.

The example metrics illustrated in the diagrams throughout this section were extracted from the Metrics Superset through a series of meetings and discussions with systems engineering personnel.

4.1.1 Purpose

The illustrated metrics along with the methodology described in this guidebook provide an example for implementing metrics on new or existing programs.

4.1.2 Systems Engineering Metrics Tree

The following diagrams provide an overview of the systems engineering example metrics and are intended to characterize SE functions.
Figure 4.1.2-1. Systems Engineering Metrics Tree (Sheet 1 of 5)
Figure 4.1.2-1. Systems Engineering Metrics Tree (Sheet 2 of 5)
Subgoal 2

Risk Tracking and Monitoring

Question(s)

SG2 – Question 1: Are risks identified early in the program, tracked and monitored throughout?

Metric: (P&C & TPM)
- Risk Assessment/Impact
- Technical Performance Measurement

Measures:
- Initial Risk Assessment
- Risk Assessments (as needed)
- TPMs (product unique)

Figure 4.1.2-1. Systems Engineering Metrics Tree (Sheet 3 of 5)
Subgoal 3  
Monitor Technical Changes & Stability

Question(s)

SG3 – Question 1: Are the requirements fully defined, stable and traceable?
Metric: (EP&Q)  
- System Requirements Statistics, Effectivity, Requirements Coverage & Deviation
Measures:
- Total Num of Reqmts
- Total Num of Reqmts Defined per Reqmt Type
- Num of TBD Reqmts per TBD Type
- Num of Derived Reqmts
- Num of Allocated Reqmts
- Num of I/F Reqmts per I/F Type
- Num of Reqmts Allocated per Reqmt Type by Phase
- Num of System Spec Effectivity Reqmts (total)
- Num of System Spec Effectivity Reqmts in ORS Document
- Num of ORS Reqmts not in System Spec (effectivity)

SG3 – Question 2: What types of requirements changes are made throughout the project’s life?
Metric: (EP&Q & TPM)  
- System Requirements Changes
Measures:
- Num of Reqmts Changes (per Reqmts Category)
- Requirements Stability (# Chgd Reqmts / # Reqmts)
- TPMs

SG3 – Question 3: How many customer initiated or internally initiated changes are made that might affect the technical stability and quality of the product/system?
Metric: (EP&Q)  
- Engineering Change Prop/Order (ECP/ECO) Statistics
Measures:
- Num of ECPs due to Customer
- Num of ECPs due to Contractor
- Number of ECPs/ECOs - phase

SG3 – Question 4: What is the length of time and cost required to prepare each ECP/ECO?
Metric: (EP&Q)  
- Engineering Change Prop/Order (ECP/ECO) Statistics
Measures:
- Cost ($$$) to Prepare each ECP or Major ECO
- Effort (LH) to Prepare each ECP or Major ECO

SG3 – Question 5: What is the length of time and cost required to implement each ECP/ECO?
Metric: (EP&Q)  
- Engineering Change Prop/Order (ECP/ECO) Statistics
Measures:
- Estimated Effort (LH) to Implement ECP or Major ECO
- Estimated Cost ($$$) to Implement ECP or Major ECO
- Actual Effort (LH) to Implement ECP
- Actual Cost ($$$) to Implement ECP
- Estimated Schedule Impact
- Actual Schedule Impact

SG3 – Question 6: How is the status of the documentation completion monitored?
Metric: (EP&Q)  
- System Design Status
Measures:
- Number of Specifications (% Tree) Produced

Figure 4.1.2-1. Systems Engineering Metrics Tree (Sheet 4 of 5)
Subgoal 4

Reduction in Cost of Poor Quality Metric

SG4 – Question 1:
Are products and processes reviewed throughout the project to detect problems early on that might affect quality of the overall system end product? If so, by what mechanism are system level problems captured, tracked and solved?

Metric: (EP&Q)
- Sys/Sw Problem Chg Reports (SP/CRs) Statistics
- System Design Status

Measures:
- Num of Open SP/CRs (per SP/CR Type) per Mission
- Num of Closed SP/CRs (per SP/CR Type) per Mission Priority
- Num of Open Reqmts/Design Review Action Items
- Num of Closed Reqmts/Design Review Action Items
- Num of Action Items from Walkthroughs

SG5 – Question 2:
When hardware errors are discovered, by what mechanism are these problems formally identified, tracked and solved?

Metric: (EP&Q)
- Hardware Problem/Chg Reports (HP/CRs) Statistics
- Quality Trouble Report (SCNs/DNs, etc) Statistics

Measures:
- Num of Open HP/CRs (per HP/CR Type) per Category
- Num of Closed HP/CRs (per HP/CR Type) per Category
- Num of Open HP/CRs (per HP/CR Type) per Mission Priority
- Num of Closed HP/CRs (per HP/CR Type) per Mission Priority
- Num of Open QA/TPRs
- Num of Closed QA/TPRs

SG5 – Question 3:
When test errors occur, by what mechanism are these problems formally identified, tracked and solved?

Metric: (EP&Q)
- Test Problem/Chg Reports (TP/CRs) Statistics

Measures:
- Num of Open TP/CRs (per TP/CR Type) per Category
- Num of Closed TP/CRs (per TP/CR Type) per Category
- Num of Open QA/TPRs
- Num of Closed QA/TPRs

SG5 – Question 4:
During quality checks, reviews, or audits, when errors are found, by what mechanism are these errors formally identified, tracked and solved?

Metric: (EP&Q)
- Quality Trouble Report (SCNs/EDNs, etc) Statistics

Measures:
- Num of Open QA/TPRs (per QA/TPR Type)
- Num of Closed QA/TPRs (per QA/TPR Type)
- Num of Open QA/TPRs (per QA/TPR Type) per Category
- Num of Closed QA/TPRs (per QA/TPR Type) per Category
- Num of Open QA/TPRs (per QA/TPR Type) per Mission Priority
- Num of Closed QA/TPRs (per QA/TPR Type) per Mission Priority

Figure 4.1.2-1. Systems Engineering Metrics Tree (Sheet 5 of 5)
4.1.3 Example Metrics for Systems Engineering

The following pages expand on some of the metrics and measures identified to be tailored and implemented on programs where applicable. (See Chapter 7 for a discussion of tailoring.)

The examples that follow, in this section and the remainder of Chapter 4, may reference activities or life cycle phases. Development phases differ in title and content from industry to industry. The table below is intended to provide a general reference to phases in product development, without restricting terminology to a specific development standard or industry. The examples in the guidebook are not intended to advocate a specific life cycle but are to show measurement throughout product development.
### Definitions

**Cost in Labor Hours (LH) and Dollars ($$)**

The following measures represent some example cost metrics selected for systems engineering. These measures can be collected at a system, product, or program level during any phase of the product’s life cycle development. Figures 4.1.4-1 and 4.1.4-2 illustrate two examples.

1. Estimated LH per Activity [Budgeted Cost of Work Scheduled (BCWS)]
2. Actual LH per Activity [Actual Cost of Work Performed (ACWP)]
3. Threshold LH per Activity
4. Budgeted Cost of Work Performed (LH) per Activity (BCWP)
5. Estimated $$ per Activity
6. Actual $$ per Activity
7. Threshold $$ per Activity

<table>
<thead>
<tr>
<th><strong>Aerospace Model</strong></th>
<th><strong>Commercial Model</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>AOC-SDR: Award of Contract through System Design</td>
<td>Product Development Decision, Initial Design</td>
</tr>
<tr>
<td>SDR-PDR: System Design through Preliminary Design Review (includes Requirements Reviews)</td>
<td>Preliminary Design</td>
</tr>
<tr>
<td>PDR-CDR: Preliminary Design through Critical or Detailed Design Review</td>
<td>Prototype Development</td>
</tr>
<tr>
<td>CDR-CUT/HWCI/CSCI: HWCI/CSCI Development, Test</td>
<td>Product Development</td>
</tr>
<tr>
<td>CUT/HWCI/CSCI-IT&amp;T: HWCI/CSCI Integration and Test</td>
<td>Product Integration and Test</td>
</tr>
<tr>
<td>System Test</td>
<td>System and Product Field Test</td>
</tr>
</tbody>
</table>
Cost at Completion

The following metrics may be collected at product levels (CSCI or HWCI) and at the program level. Associated graphs should be plotted for each product or component and for the program as a whole. Figure 4.1.4-3 shows an example of the graphical views provided for this metric.

1. Budgeted at Completion (BAC) ($$) per Phase
2. Estimated at Completion (EAC) ($$) per Phase

Effort in LH per Labor Category

Each of the following metrics can be collected at the component level and at the program level. Associated graphs should be plotted for each graph and for the program as a whole. Labor categories within the LH can be used for analysis. An example of labor categories is shown.

- Experienced: Minimum of three years in similar applications
- Special: Specialized abilities defined as crucial to the success of the particular system.
- Total: Sum of Experienced and Inexperienced. Special are counted within the broad categories of experienced and inexperienced but are also tracked separately.

1. Planned LH per Labor Category
2. Actual LH per Labor Category
Analysis

1. Compare the measurements of cost in LH with other measurements in LH that are made at different points in time, such as with previous projects. Measurements made in dollars are difficult to compare through time or among development projects done by different organizations because of variability in labor rates (in dollars) among organizations, the value of the dollar, and the changing cost of labor.
2. A diverging plot trend on the BCWP/ACWP graph and an upward EAC plot trend indicate that costs are increasing. This could be caused by an increase in the product size or complexity, difficulty with requirements or design, difficulty during integration or testing, or personnel turnover.
3. When the EAC runs less than BAC early in the project, the usual explanation is that staffing is not proceeding as quickly as planned. This can also result from including an overbudgeted level of effort (LOE) task in the results; removing this from the metric will give a better indication of tracking to the plan by the milestone tasks.
4. Cost indicators are useful for historical purposes to compare bid estimates with the actuals and calibrate bid costs.

4.2 Test Engineering Measurement Overview

This section provides an overview of the example test engineering metrics, their relationship and support to an organization's goals, and selected metric examples.

The example metrics illustrated in the diagrams throughout this section were extracted from the Metrics Superset through a series of workshops and discussions with test engineering personnel. During the workshops, some of the metrics selected from the superset were modified to meet the intent of the collection process, and some new metrics were added.

4.2.1 Purpose

The illustrated metrics along with the methodology described in this guidebook provide an example for implementing metrics on a new or existing program.
4.2.2 Test Engineering Metrics Tree

The following diagrams provide an overview of the test engineering metrics and their support of various organizational goals.
Figure 4.2.2-1a. Test Metrics Tree (Sheet 2 of 5)
Subgoal 2

Question(s)

SG2 – Question 1: Are the requirements fully defined, stable and traceable?

Metric: (EP&Q)
- Test Requirements Coverage & Deviation
-Reqmts Test Procedures Status (includes Derived)

Measures:
- Num of Test Reqmts
- Num of Derived Test Reqmts
- Num of System Spec Test Reqmts (total)
- Num of System Spec Test Reqmts in Test Plan
- Act Num of Test Reqmts Allocated to Test Procedure per Phase
- Act Num of Reqmts per Test

SG2 – Question 2: How many test procedures are written as compared to those that need to be rewritten or changed?

Metric: (EP&Q)
- Reqmts Test Procedures Status (includes Derived)

Measures:
- Planned Num of Test Procedures (in Test Plan)
- Act Num of Test Procedures Written by Phase
- Actual Num of Test Procedures Approved
- Actual Num of Test Procedures Rewritten per Phase
- Actual Num of Test Procedures Rewritten per Phase based on Design Changes before Test Execution (log input)

SG2 – Question 3: How many test procedures are dry run?

Metric: (EP&Q)
- Reqmts Test Procedures Status (includes Derived)

Measures:
- Planned Num of Test Procedures to be Dry Run
- Act Num of Test Procedures Dry Run

SG2 – Question 4: How many test procedures are formally tested?

Metric: (EP&Q)
- Reqmts Test Procedures Status (includes Derived)

Measures:
- Planned Num of Test Procedures Formally Tested
- Actual Num of Test Procedures Formally Tested

Figure 4.2.2-1. Test Metrics Tree (Sheet 3 of 5)
Subgoal 3: Reduction in Cost of Poor Quality

Question(s)

SG3 – Question 1: When test errors occur, by what mechanism are these problems formally identified, tracked and solved?

Metric: (EP&Q) - Test Problem/Chg Reports (TP/CRs) Statistics

Measures:
- Num of Open TP/CRs (per TP/CR Type) per Category
- Num of Closed TP/CRs (per TP/CR Type) per Category
- Num of Open TP/CRs (per TP/CR Type) per Mission Priority
- Num of Closed TP/CRs (per TP/CR Type) per Mission Priority

SG3 – Question 2: What kinds of test errors are discovered during various stages of development?

Metric: (EP&Q) - Test Problem/Chg Reports (TP/CRs) Statistics

Measures:
- Num of TP/CRs per Project/Controlled Phase (per TP/CR Type)

SG3 – Question 3: What is the length of time it takes to fix a problem/error once discovered?

Metric: (EP&Q) - Test Problem/Chg Reports (TP/CRs) Statistics

Measures:
- Duration of time Each TP/CR (per TP/CR Type) Open (log)

SG3 – Question 4: What is the associated cost (LH) to fix the error/problem?

Metric: (EP&Q) - Test Problem/Chg Reports (TP/CRs) Statistics

Measures:
- Act Effort (LH) to Fix each TP/CR (per TP/CR Type)

SG3 – Question 5: Are test requirements reviewed for consistency and completeness?

Metric: (EP&Q) - Test Requirements Discrepancies

Measures:
- Num of Reqmts Discrepancies per (contractual) Review
- Num of Open Reqmts Discrepancies
- Num of Closed Reqmts Discrepancies

Figure 4.2.2-1. Test Metrics Tree (Sheet 4 of 5)
Subgoal 4

Monitor Testing Activities

Question(s)

SG4 – Question 1: What is the level of requirements validation?

Metric: (EP&Q) - Test Coverage

Measures:
- Act Num of Reqmts Validated by Test
- Act Num of Reqmts Validated by Demo
- Act Num of Reqmts Validated by Inspection
- Act Num of Reqmts Validated by Analysis
- Num of Tests Passed
- Num of Tests Failed
- Act Num of Test Executions of each Test to Meet Pass/Fail Criteria (step level - test log)

SG4 – Question 2: What is the effort (LH) required for validation?

Metric: (EP&Q) - Test Coverage

Measures:
- Act LH Expended to Execute Test & Analyze Results

SG4 – Question 3: What is the overall test success ratio?

Metric: (EP&Q) - Breadth of Testing

Measures:
- Test Coverage (# Reqs Tested/Total # Reqs)
- Test Success (# Reqs Passed/# Reqs Tested)
- Overall Success (# Reqs Passed/Total # Reqs)

SG4 – Question 4: Once the system is at site, how are the test activities monitored and tracked?

Metric: (EP&Q) - Site Testing

Measures:
- Act Num of SP/CR Changes during Site Testing
- Act Num of New Tests Conducted During Site Testing
- Act Num of New Test Procedures Written during Site Testing
- Act Num of New Test Procedures Revised during Site Testing
- Percentage of Site Configuration Changes Resulting in Retesting (redlines, out-of-scope, ECPs/RFCs)

Figure 4.2.2-1. Test Metrics Tree (Sheet 5 of 5)

4.2.3 List of Core Metrics for Test Engineering

The following pages expand on some of the metrics and measures identified by the Test Engineering Working Groups to be tailored and implemented on programs.

The following are examples of test measures/metrics.

Technical Stability Indicators

Staffing is one of the metrics that plays an important role in the technical stability of a program. Staffing metrics show when a program has insufficient human resources and can be displayed as shown in Figure
4.2.4-1. Insufficient schedule and budget are also related to technical instability, but these situations are covered by other management indicators.

**Definition**

**Staffing Profiles**

Each of the following metrics will be collected for each component. Associated graphs should be plotted for each component and for the program or product as a whole. The first two measures are identified as core metrics for test engineering.

1. Estimated Staffing Profile per Labor Category
2. Actual Staffing Profile per Labor Category
3. Unplanned Staff Losses per Labor Category
4. Unplanned Staff Gains per Labor Category

![Staffing Profiles Graph](image)

**Figure 4.2.4-1. Staffing Profiles**

**Analysis**

1. Technical stability of the development project is important because stability is often the prime determinant of whether or not the project will be completed within the required cost, schedule, and quality requirements imposed upon it. It is often difficult to find a sufficient number of qualified personnel, and a shortage of such people can severely impact the schedule.
2. The shape of the experienced staff profile in an IPT environment is often high during the initial stage of the project, dipping slightly during development and then growing during testing. An example ratio of total to experienced personnel is 3:1 with a cap of 6:1. Insufficient experienced personnel may lead to poor quality, which will cause schedule slips in later phases.
3. Understaffing results in schedule slippage and, if not corrected, in a continuing rate of slippage.
4. Adding inexperienced staff to a late project will seldom improve the schedule and often causes further delays. Adding staff with the relevant experience may improve the schedule. However, it may be difficult to staff up with more experienced personnel in the required time, and gains in the far future (more trained personnel) may outweigh near term schedule problems. In either case, cost will increase.
5. A program that is experiencing a high personnel turnover rate cannot maintain needed continuity. Losses that impair the project knowledge and experience base should be treated as project risks with cost and schedule impacts.
6. A sudden increase in unplanned personnel losses may indicate the onset of a morale problem.

4.3 Software Engineering Measurement

This section provides an overview of the software engineering example metrics, their relationship and support to an organization's goals, and selected metric examples.

The example metrics illustrated in the diagrams throughout this section were extracted from the Metrics Superset through a series of workshops and discussions with systems software personnel. During the workshops, some of the metrics selected from the superset were modified to meet the intent of the collection process, and some new metrics were added.

4.3.1 Purpose

The illustrated metrics along with the methodology described in this guidebook provide an example for implementing metrics on a new or existing program.

4.3.2 Software Engineering Metrics Tree

The following diagrams provide an overview of the example software engineering metrics set and their support of various organizational goals.
Metrics Guidebook for Integrated Systems and Product Development

Figure 4.3.2-1. Software Metrics Tree (Sheet 1 of 4)
Subgoal 1
Evaluate & Track
Productivity
Subgoal 1

Question(s)

SG1 – Question 1: What size is the developed software units?
Metric: (P&C)
Source Lines of Code (SLOC) Count
Measures:
- Estimated SLOC per Code Type
- Actual SLOC per Code Type
- Est Num of HCI Displays
- Actual Num of HCl

SG1 – Question 2: What is the productivity for completed projects?
Metric: (P&C)
- Months in Schedule
- Major Milestones
- Detailed Milestones
Measures:
- Planned Calendar Months per Phase
- Actual Calendar Months per Phase
- Estimated Dates of Major Milestones
- Actual Dates of Major Milestones
- Slack of Major Milestones
- Initial Num of Detailed (Tasks) Milestones - Plan
- Actual Detailed (Task) Milestones Completed

SG1 – Question 3: What is the degree of change? (i.e. proportion of defined units to new or deleted units)
Metric: (P&C)
Software (CSU & CSC) Status
Measures:
- Projected Number of CSUs
- Actual Number of CSUs
- Projected Number of CSCs
- Actual Number of CSCs

SG1 – Question 4: In which programming languages are the components to be changed written?
Metric: (P&C)
Source Lines of Code (SLOC) Count
Measures:
- SLOC Total per Language

SG1 – Question 5: What is the effort spent for development (in LH)?
Metric: (P&C)
- Cost in LH (per WBS activities)
- Effort in LH
Measures:
- Estimated LH per Activity (BCWS)
- Actual LH per Activity (ACWP)
- Planned LH per Labor Category
- Actual LH per Labor Category

Figure 4.3.2-1. Software Metrics Tree (Sheet 2 of 4)
Subgoal 2

Monitor Quality

Question(s)

SG2 – Question 1: Are the requirements fully defined, stable and traceable?

Metric: (EP&Q) -Software Requirements Coverage & Deviation

Measures:
-Cumulated Num of Req
-Allocated to Sw
-Num of Changes to Allocated Req
-Num/% of System Spec Software Req
-Num/% of Derived Software Req
-Num/% of Sys Spec Sw
-Reqs in SRS & IRS
-Num/% of SRS & IRS Req
-Num/% of SRS & IRS Req not in Sys Spec (Sw)
-Num/% of Sw Req
-Track through Completion of Acceptance Testing

SG2 – Question 2: How many residual errors are still in the system when delivered?

Metric: (EP&Q) -Sw Problem Chg Reports (SP/CRs) Statistics

Measures:
-Num of Open SP/CRs (per SP/CR Type)
-Num of Closed SP/CRs (per SP/CR Type)

SG2 – Question 3: What types of errors are discovered during various stages of development?

Metric: (EP&Q) -Sw Problem Chg Reports (SP/CRs) Statistics

Measures:
-Num of SP/CRs per Project/Controlled Phase (per SP/CR Type)
-Num of SP/CRs per Project/Controlled Phase (per Category)
-Num of Open SP/CRs (per SP/CR Type) per Category
-Num of Closed SP/CRs (per SP/CR Type) per Category
-Num of Open SP/CRs (per SP/CR Type) per Mission Priority
-Num of Closed SP/CRs (per SP/CR Type) per Mission Priority

SG2 – Question 4: How long does it take to fix an error once discovered?

Metric: (EP&Q) -Sw Problem Chg Reports (SP/CRs) Statistics

Measures:
-Duration of time Each SP/CR (per SP/CR Type) Open (log)

SG2 – Question 5: What is the associated cost (in LH) to fix a discovered error?

Metric: (EP&Q) -Sw Problem Chg Reports (SP/CRs) Statistics

Measures:
-Actual Effort (LH) to Fix Each SP/CR (per SP/CR Type)

Figure 4.3.2-1. Software Metrics Tree (Sheet 3 of 4)
Subgoal 3  
Detect Impediments to Productivity

Question(s)

SG3 – Question 1: Are products and processes inspected throughout the project's lifecycle to detect problems early on that might affect productivity later?

Metric: (EP&Q) -Inspection - Defect Profiles

Measures:
- Actual Number of Major Defects per Defect Type per Phase
- Actual Number of Major Missing Defects
- Actual Number of Major Extra Defects
- Actual Number of Minor Defects per Defect Type per Phase
- Actual Number of Minor Wrong Defects
- Actual Number of Minor Missing Defects
- Actual Number of Minor Extra Defects

 SG3 – Question 2: How many errors are there?

Metric: (EP&Q) -Inspection - Defect Profiles

Measures: (Same measures used answering each of questions 1 thru 4)

 SG3 – Question 3: When are the errors found?

Metric: (EP&Q) -Inspection - Defect Profiles

Measures: (Same measures used answering each of questions 1 thru 4)

 SG3 – Question 4: What type of errors are being found?

Metric: (EP&Q) -Inspection - Defect Profiles

Measures: (Same measures used answering each of questions 1 thru 4)

 SG3 – Question 5: How effective are inspections in finding critical as well as minor problems early on in the project?

Metric: (EP&Q) -Inspection - Effectiveness

Measures:
- Actual Number of Major Defects per Inspection
- Actual Number of Minor Defects per Inspection

Figure 4.3.2-1. Software Metrics Tree (Sheet 4 of 4)
4.3.3 Example Metrics for Software Engineering

The following pages expand on some of the metrics and measures identified by software engineering working groups, to be tailored and implemented on programs where applicable. The Metrics are the same previously shown in section 4.4.2 diagrams, illustrating the relationship between goals, questions, metrics, and measures.

The following are examples of measures/metrics in the software example metrics set.

Product Size Indicator

This metric tracks the size of the software product relative to estimate. Tracking of the actual code counts should be done whenever actual code counts are available (from code and unit test until the end of the project). During software product in design, tracking of code can be accomplished by converting the counts of source lines of design, process bubbles, or objects to estimates of source lines of code.

Definition

Source Lines of Code (SLOC) Count

Source Lines of Code counts are collected per Code Type (such as new, modified, added, changed, deleted, etc.).

The following metrics are collected for each CSU, CSC, CSCI and Program.

Product Size Indicators

1. Estimated SLOC per Code Type
2. Actual SLOC per Code Type

Equivalent to new Source Lines Of Code (ESLOC) Count

The ESLOC metric combines all of the Code Types of source statements into one size metric.

The ESLOC concept allows computation of overall size and labor rate of productivity metrics for software products that include both new and reused code. The ESLOC concept facilitates size and productivity comparisons among computer programs and among development projects with varying code mixtures.

Another example of measures that can be collected for each CSCI.

1. Estimated ESLOC
2. Actual ESLOC
### Analysis

1. Knowledge of the size of the planned software product is important because the size, however measured, will be one of the main determinants of resource allocation (dollars and personnel) to the development of the software product.

2. There may be memory or CPU processor constraints that restrict program size, and these restrictions can become critical in the late stages of development. (TPM)

3. SLOC is initially an estimate that should become more and more accurate until it represents the actual code at completion.

4. SLOC estimate increases may be due to additional requirements, to a better understanding of the requirements or of the design, or to an optimistic original estimate. Increases in size should be accounted for in the project’s schedule and staffing plans.

5. SLOC estimate decreases usually result from an overestimate at the beginning of the program and not from changes in requirements.

6. Modifying and even removing SLOC from code not developed by the team will impact both the documentation and testing effort. When greater than 40 percent of a module slated for reuse needs to be modified or removed, it may cost less to develop the code as new.

7. The ESLOC concept allows you to compute overall size and labor rate or productivity for software products that include both new and reused code. The ESLOC concept facilitates size and productivity comparisons among computer programs and among development projects with varying code mixtures. *(Note: For productivity comparisons performed, the data needs to be normalized by considering all attributes of the program or project.)*

8. These size metrics are useful for historical purposes to compare bid estimates with the actuals and calibrate our bid costs.

#### Figure 4.3.4-1. Software Product Size

<table>
<thead>
<tr>
<th>Months Into Program</th>
<th>New Estimate</th>
<th>New Actual</th>
<th>Reused Estimated</th>
<th>Reused Actual</th>
<th>Modified Estimated</th>
<th>Modified Actual</th>
<th>Total Estimated</th>
<th>Total Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **KSLOC**
4.4 Hardware Engineering Measurement Overview

This section provides an overview of the example hardware engineering metrics, their relationship and support to an organization’s goals, and selected metric examples.

The metrics illustrated in the diagrams throughout this section are extracted from the Metrics Superset through a series of workshops and discussions with hardware engineering personnel. During the workshops, some of the metrics selected from the superset were modified to meet the intent of the collection process, and some new metrics were added.

4.4.1 Purpose

The illustrated metrics along with the methodology described in this guidebook provide an example for implementing metrics on a new or existing program.

The following diagrams provide an overview of the example hardware engineering metrics and their support of various organizational goals.
Primary Management Goals

Implement More Effective Hardware Proposal Costing/BOE and ROM Processes

Realize a 20% Reduction in Turnaround Time for Costings

Lead/Support IPD Process Action Team Goals to Achieve Improvements in IPD's Processes/ Business Practices

Improve Processes to Reduce Cost of Poor Quality While Decreasing Turnaround Time

Subgoals

Substantiate Pertinent Sizing Input Categories Through Accumulation of Trend Data Subgoal 1

Substantiate Pertinent Costing Input Categories Through Accumulation of Trend Data Subgoal 2

Monitor Quality during Project Development Subgoal 3

Monitor Technical Stability/ Changes Subgoal 4

Monitor Cycle Time Subgoal 5

Sheets 2 and 3

Sheets 4

Sheet 5

Sheets 7 and 8

Sheets 6

Subgoal 1

Subgoal 2

Subgoal 3

Subgoal 4

Subgoal 5

Figure 4.4.1.1. Hardware Metrics Tree (Sheet 1 of 8)
Subgoal 1: Substantiate Pertinent Sizing Input Categories Through Accumulation of Trend Data

Question(s)

SG1 – Question 1: What is the variant between estimates and actuals for various types of circuit card assemblies (CCAs)?

Metric: (EP&Q) - Circuit Card Technologies (per category & type)

Measures:
- CCA Digital - Number/ % New Design
- CCA Analog - Number/ % New Design
- CCA Mixed - Number/ % New Design
- CCA RF - Number/ % New Design

SG1 – Question 2: What percentage of CCAs require new design?

Metric: (EP&Q) - Circuit Card Technologies (per category & type)

Measures:
- CCA Digital - Number/% New Design
- CCA Analog - Number/% New Design
- CCA Mixed - Number/% New Design
- CCA RF - Number/% New Design

SG1 – Question 3: What percentage of CCAs require modification?

Metric: (EP&Q) - Circuit Card Technologies (per category & type)

Measures:
- CCA Digital - Number Modified
- CCA Analog - Number Modified
- CCA Mixed - Number Modified
- CCA RF - Number Modified

SG1 – Question 4: What percentage of CCAs require off-the-shelf components?

Metric: (EP&Q) - Circuit Card Technologies (per category & type)

Measures:
- CCA Digital - Number/% COTS
- CCA Analog - Number/% COTS
- CCA Mixed - Number/% COTS
- CCA RF - Number/% COTS

SG1 – Question 5: What is the variant between estimates and actuals for various types of Chassis/Racks?

Metric: (EP&Q) - Chassis/Racks Measures

Measures:
- Estimated Number of Chassis (by type)
- Actual Number of Chassis (by type)
- Estimated Number of Racks (by type)
- Actual Number of Racks (by type)

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 2 of 8)
Subgoal 1: Substantiate Pertinent Sizing Input Categories Through Accumulation of Trend Data

Question(s)

SG1 – Question 6: What percentage of Chassis/Racks require new design?
Metric: (P&C) - Chassis/Racks Measures
Measures:
- Number/% Chassis - New Design
- Number/% Racks - New Design

SG1 – Question 7: What percentage of Chassis/Racks require modification?
Metric: (P&C) - Chassis/Racks Measures
Measures:
- Number of Modified Chassis
- % Modification to Chassis
- Number of Modified Racks
- % Modification to Racks

SG1 – Question 8: What percentage of Chassis/Racks require off-the-shelf components?
Metric: (P&C) - Chassis/Racks Measures
Measures:
- Number/% COTS Used - Chassis
- Number/% COTS Used - Racks

SG1 – Question 9: What is the variant between estimates and actuals for various types of ASICs to be developed?
Metric: (P&C) - ASIC Measures
Measures:
- Estimated Number of ASICs to be Developed (by type)
- Actual Number of ASICs to be Developed (by type)

SG1 – Question 10: What is the variant between estimates and actuals for Firmware LOC Count?
Metric: (P&C) - Firmware Count Measures
- Estimated LOC Per Code Type (new, mod, etc.)
- Derived LOC per Code Type (fudge factor)
- Actual LOC per Code Type
- Estimated Total LOC per Language
- Actual Total LOC per Language

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 3 of 8)
Subgoal 2

Substantiate Pertinent Costing Input Categories Through Accumulation of Trend Data

Question(s)

SG2 – Question 1:
What is the effort spent for development (in LH)?

Metric: (P&C)
-Cost in LH (per WBS activities)

Measures:
-Estimated LH per Activity (BCWS)
-Actual LH per Activity (ACWP)
-BCWP (LH) per Activity
-Variance (LH) per Activity (schedule & cost)

SG2 – Question 2:
What is the effort spent for development (in $$)?

Metric: (P&C)
-Cost in $$ (per WBS Activities-Plng, etc.)

Measures:
-Estimated Cost ($$) per Activity
-Actual Cost ($$) per Activity

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 4 of 8)
Subgoal 3

Monitor Quality during Project Development

Question(s)

SG3 – Question 1: How many residual errors are still in the system when delivered?

Metric: (EP&Q)
-Hw Problem/Chg Reports (HP/CRs) Statistics

Measures:
- Num of Open HP/CRs (per HP/CR Type)
- Num of Closed HP/CRs (per HP/CR Type)

SG3 – Question 2: What types of hardware errors are discovered during various stages of system development?

Metric: (EP&Q)
-Hw Problem/Chg Reports (HP/CRs) Statistics

Measures:
- Num of HP/CRs per Project/Controlled Phase (per HP/CR Type)
- Num of Open HP/CRs (per HP/CR Type) per Category
- Num of Closed HP/CRs (per HP/CR Type) per Category
- Num of Open HP/CRs (per HP/CR Type) per Mission Category
- Num of Closed HP/CRs (per HP/CR Type) per Mission Category

SG3 – Question 3: How long does it take to fix an error once discovered?

Metric: (EP&Q)
-Hw Problem/Chg Reports (HP/CRs) Statistics

Measures:
- Duration of time each HP/CR (per HP/CR Type) Open (log)

SG3 – Question 4: What is the associated cost (in LH) to fix an error?

Metric: (EP&Q)
-Hw Problem/Chg Reports (HP/CRs) Statistics

Measures:
- Estimated effort (LH) to fix each HP/CR per HP/CR Type
- Effort (LH) to fix each HP/CR (per HP/CR Type)

SG3 – Question 5: What is the associated cost (in $$) to fix an error?

Metric: (EP&Q)
-Hw Problem/Chg Reports (HP/CRs) Statistics

Measures:
- Estimated Cost ($) to fix each HP/CR per HP/CR Type
- Cost ($) to fix each HP/CR per HP/CR Type

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 5 of 8)
Subgoal 4

Monitor Technical Stability/Changes

Question(s)

SG4 – Question 1: How many internal changes are initiated on each project?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Number of ECOs per Document Released - Total Snapshot
- Number of ECOs vs Number of Drawings

SG4 – Question 2: How long does it take to complete each ECO (from initiation to finish)?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Length of ECO Cycle Time (average)

SG4 – Question 3: What is the length of time and cost required to analyze each ECO?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Effort (LH) to analyze each ECP or Major ECO
- Cost ($) to analyze each ECP or Major ECO

SG4 – Question 4: What is the length of time and cost required to prepare each ECO?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Effort (LH) to prepare each ECP or Major ECO
- Cost ($) to prepare each ECP or Major ECO

SG4 – Question 5: What is the length of time to implement each ECO/ECP?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Effort (LH) to implement each ECP or Major ECO
- Cost ($) to implement each ECP or Major ECO

SG4 – Question 6: What is the cost ($) to test each incorporated ECO/ECP?

Metric: (EP&Q)
- Engineering Change Prop/Order (ECP/ECO) Status

Measures:
- Estimated Cost ($) to test each major incorporated change
- Actual Cost ($) to test each major incorporated change

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 6 of 8)
Subgoal 5

Monitor Cycle Time

Question(s)

SG5 – Question 1: How long does it take to design and fabricate various types of CCAs by level of complexity?

Metric: (EP&Q) - Hardware Status

Measures:
- Estimated time to des/fab [simple, moderate, complex] CCAs (by type)
- Actual time to des/fab [simple, moderate, complex] CCAs (by type)

SG5 – Question 2: How many iterations or cycles are required per electronic circuit?

Metric: (EP&Q) - Hardware Status

Measures:
- Estimated design cycles per electronic circuit (by type)
- Actual design cycles performed per electronic circuit (per type)

SG5 – Question 3: What are the total number of drawings completed vs estimated?

Metric: (EP&Q) - Hardware Status

Measures:
- Projected Number of Drawings
- Actual Number of Drawings completed

SG5 – Question 4: How long does it take to design and fabricate PCBs?

Metric: (EP&Q) - Hardware Design Process Efficiency

Measures:
- Planned PCB Design Cycle Time
- Actual PCB Design Cycle Time
- Planned PCB Fabrication Cycle Time
- Actual PCB Fabrication Cycle Time

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 7 of 8)
Subgoal 5

Monitor Cycle Time

Question(s)

SG5 – Question 5: What is the average time for fabrication of Sheet Metal?

Metric: (EP&Q)
-Hardware Design
Process Efficiency

Measures:
-Planned Sheet Metal Fabrication Cycle Time
-Actual Sheet Metal Fabrication Cycle Time

SG5 – Question 6: How long does it take to order and receive parts?

Metric: (EP&Q)
-Hardware Design
Process Efficiency

Measures:
-Planned Parts Ordering cycle time
-Actual Parts Ordering cycle time

SG5 – Question 7: Are tools used to help reduce cycle time and increase efficiency in the hardware process?

Metric: (EP&Q)
-Hardware Design Tool Usage Measures

Measures:
-Est # simulation cycles per circuit designed
-Act # simulation cycles per circuit designed
-Est # simulation hours per circuit designed
-Act # simulation hours per circuit designed
-Est # Mechanical Library Elements used per new mechanical drawing
-Act # Mechanical Library Elements used per new mechanical drawing
-Est # Electrical Library Elements used per new electrical schematic
-Act # Electrical Library Elements used per new electrical schematic
-% Concurrent CAE/CAD Sw License Used

Figure 4.4.1-1. Hardware Metrics Tree (Sheet 8 of 8)
4.4.2 Example Metrics for Hardware Engineering

The following pages expand on some of the metrics and measures identified by the Hardware Engineering Working Groups to be tailored and implemented on programs where applicable.

Quality Indicators

Quality of the hardware and of the hardware development process is an important consideration, and the basic quality measure is defects. To produce a high-quality product, the development process should be attuned to a zero-defect method of operation. To produce a high-quality product at the lowest possible cost, remove defects at the earliest possible point in the development process. Although testing with the related metrics is important, inspections and reviews and the metrics related to inspections and reviews are also important.

Definition

Hardware Problem Change Report (HP/CR) Statistics

The HP/CRs are written on items before or after configuration control during the following phases:

- Component testing
- HWCI testing
- System testing

The HP/CRs are tracked by priority and category.

Priority identifies the priority of this HP/CR with respect to product functionality. This priority is tracked as follows.

1. causes essential function to be disabled or jeopardizes personnel safety.
2. causes essential function for which there is no workaround to be degraded.
3. causes essential function for which there is a reasonable workaround to be degraded.
4. causes user inconvenience but doesn't affect an essential function.
5. all other errors.

Category identifies the area in which the problem lies for the HP/CR.

- Hardware: Hardware is not sufficient to allow operations according to supporting documentation, and the documentation is correct.
- Documentation: Hardware does allow sufficient operation according to supporting documentation, but the documentation is incorrect.
- Design: Hardware operates according to supporting documentation, but a design deficiency exists.
- Requirements: Requirements deficiency exists.

Several of the following metrics can be collected for every hardware product. Associated graphs should be plotted for each product and for the system as a whole. Figures 4.4.2-1, 4.4.2-2, and 4.4.2-3 show examples of the graphical views provided for this metric.
1. Number of HP/CRs per Controlled Phase
2. Number of Open HP/CRs
3. Number of Closed HP/CRs
4. Number of Open HP/CRs per Mission Priority
5. Number of Closed HP/CRs per Mission Priority
6. Number of Open HP/CRs per Category
7. Number of Closed HP/CRs per Category
8. Length of time each HP/CR is Open
9. Effort (LH) to fix each HP/CR

Figure 4.4.2-1. Cumulative HR/CRs

Figure 4.4.2-2. Category of HP/CRs
Analysis
1. Often the plot of requirements defects or HP/CRs rises prior to and peaks at the end of the design phase and then tapers off after each review. Projects that produce clear and complete specifications will experience less of a rise at each review. HP/CRs tend to peak again during component and HWCI testing.
2. If the number of new and open HP/CRs from component Integration Testing is not approaching zero as product delivery approaches and the new problems being discovered are not minor, the product delivery should be delayed. The schedule slip may be estimated by observing the trend of new HP/CRs. Similar analysis applies to hardware testing and the start of system testing.
3. Too few HP/CRs opened during component Integration Testing may indicate poor testing and too many may indicate poor component quality.
4. A critical HP/CR will impact the schedule for subsequent activities if it remains open. The presence of past due critical or urgent HP/CRs is a sign that the project will be subject to increasing schedule slippage.

4.5 Manufacturing Engineering Measurement Overview

This section provides an overview of the example manufacturing metrics (an initial set), their relationship and support to an organization's goals, and selected metric examples.

The core metrics illustrated in the diagrams throughout this section were extracted from the Metrics Superset through a series of inputs and discussions with manufacturing personnel. During the discussions, some of the metrics selected from the superset were modified to meet the intent of the collection process, and some new metrics were added.

4.5.1 Purpose

The illustrated metrics along with the methodology described in this guidebook provide example metrics for implementation on new or existing programs.
4.5.2 Manufacturing Metrics Tree

The following diagrams provide an overview of the example manufacturing metrics and their support of various organizational goals.

![Manufacturing Metrics Tree Diagram]

**Figure 4.5.2-1. Manufacturing Metrics Tree**

4.5.3 Example Metrics for Manufacturing

The following pages expand on some of the metrics and measures identified by Manufacturing Engineering working groups as an initial example metrics set, to be tailored and implemented on programs where applicable.

The following are manufacturing example metrics.
Project Status Indicators

The following illustrates how measures can be collected to monitor and track manufacturing parts.

Definition

*Material Control Status*

1. Projected Total Number of Project Parts to be Ordered
2. Number of Parts Ordered
3. Number of Parts Received

Analysis

This metric shows a simple baseline for ordering parts. A number of parts are projected based on inputs from the pre-proposal, proposal, and other early activities. After contract award, manufacturing begins the process of ordering the identified parts. Times of ordering depend upon the schedule and the need. This chart can be used for process analysis and improvement.

1. If the parts received fall behind those ordered, it is time to look at the reasons for the shortfall. The areas to investigate will include vendor response, vendor supply, in-house ordering process, internal receiving process, and so on.
2. When the projected number of parts falls above or below the actual parts needed, another type of problem exists such as poor requirements definition, changing design, or the human factor, where lack of experience can be the problem.
3. Not shown on this chart, but often an issue in material control, is the percentage of assembly pieces delivered that are not acceptable and must be returned for rework or replacement.
4. Another input to the data are goals identified for the organization. These can be based on internal trend data or trend data available through vendors and the industry. A sample goal might be to reduce parts rejection to 2% per 1000 pieces.

![Figure 4.5.3-I. Manufacturing Parts Projected, Ordered, and Received](image-url)
4.6 TPMs for IPTs

Technical Performance Measures (TPMs) are system or product specific, quantifiable measures that express critical system or product performance parameters relative to performance requirements. TPMs may express formal or market driven requirements. TPMs are used to track those parameters that are critical to performance and at risk. This type of metric is used in design, development, and field maintenance phases, and is generally “owned” by the product IPT, though it may be monitored by a specific function. In the front-end phases of product development, TPMs are generally used to understand allocation and implications of design on system performance budgets, to manage product and technical risk. TPMs are generally used in development and field support phases to demonstrate (verify) system or product performance.

There are two levels of TPMs. The first is the system (or aggregate) level TPM that measures total product or system performance and compliance to the customer's critical needs. This set of TPMs often requires the second level, product, or component data from multiple IPTs to build a complete picture. TPMs are usually depicted using control charts or stacked column or line charts with target and hard requirement values. The timing of collection can be event driven, monthly, weekly, or even daily when following critical activities.

TPMs are the product design assessment used to estimate, through engineering analysis and test, the values of essential performance parameters of a product or system. TPMs are used to forecast the values to be achieved through the planned technical program effort, to measure differences between the achieved values and those allocated to the product element by the systems engineering process, and to determine the impact of these differences on system effectiveness. TPMs are designed to:

1. Provide visibility of actual versus planned performance.
2. Provide early detection or prediction of problems that require management attention.
3. Support assessment of the program impact of proposed change alternatives.

TPMs alert program management to potential performance deficiencies before irrevocable cost or schedule impact occurs. Where a program also has an overall risk assessment program, TPMs provide data for technical risk planning and assessment. Input from the risk management process will also assist in determining parameter criticality in the TPM selection process.

TPMs are the key measures of the characteristics of the product/system. They represent key specification parameters of the product/system. The use made of TPMs through the requirements definition, design, and development of a product or system typically include the following steps:

1. Identification of TPMs. This step is performed during the requirements definition and analysis phase. Often, TPMs represent cost drivers and/or characteristics of the desired product/system that are pushing the state of the art. TPMs are important to monitor during design and development to maintain and understand the feasibility and risk of developing the product/system so that it can actually meet the desired specifications. There are as many TPMs as there are products. Examples are:
   a. Physical Component Measures (weight, footprint, height, density, volume, etc.)
   b. Mechanical, electrical (power, battery life) and thermal measures (cooling capacity)
   c. Processing speed and size (throughput, memory, time to display)
   d. Safety (response time, boundary conditions)
   e. Reliability, availability, and maintainability
2. **Determining Expected Performance.** Figure 4.6-1 depicts a TPM profile. The TPM profile shows expected behavior (planned value) over time, compared with the required value (specification requirement). A planned value profile is developed showing the expected value versus relevant events during the program. The tolerance band indicates the acceptable limit (upper and lower), with reference to the required value.

![Figure 4.6-1. TPM Profile](image)

2. **Determining the Demonstrated Value.** As the program progresses, actual (demonstrated) values are plotted. “Demonstrated” values may be obtained by analysis simulation and/or prototyping. After fabrication, measured (tested) values using the hardware/software elements are used.

3. **Defining Current Estimate.** Using the demonstrated value(s), the current estimate is determined. The current estimate is the value of a parameter predicted for the end product/system on the production phase.

4. **Monitoring TPM Margin.** Each demonstrated value is compared against the tolerance imposed by the TPM profile. The difference between the specification requirement and the current estimate of the parameter is the predicted technical variance.
5. **Addressing Risk.** When a demonstrated value is out of tolerance, the performance of the TPM is treated as a system risk and is tracked by the risk management process. Risk mitigation may include:

   a. *Relaxation of the Requirement.* The customer or organization may decide that the requirement is not critical to mission operation and relax the requirement, which can result in significant savings in dollars and/or schedule.

   b. *Parallel Development.* Management may make the decision to start parallel development of a different approach to implement a key component/subsystem, thereby increasing significantly the chances of overall system success.

   c. *Reassignment of Personnel.* A decision to shift key personnel to a high-risk part of the system design, leaving the lower-risk parts of the system design to the less experienced personnel.

   d. *Increasing Resources.* A decision to put more resources into a portion of the system to more clearly ensure product delivery.

The development of a catalog of TPMs which are appropriate and useful for the product or services your organization provides can be a real competitive advantage. Over time, TPMs can provide a springboard for the pro-active identification of opportunities to improve processes used in product development, as well as providing an effective mechanism for allocation of priorities and resources in the design and development of a product/system to distribute risk reasonably evenly across the entire effort.

<table>
<thead>
<tr>
<th>TPMs are used not only in the context of a developing program but also provide important information about systems that affect our daily lives. As an example, TPMs have been defined by a California Water District to gauge the health of a nearby watershed area as well as the quality of the water. Some of these metrics include:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Percentage nitrates – redwood trees absorb nitrates. If the nitrate content of the water is increasing, it may indicate the local forest is declining.</td>
</tr>
<tr>
<td>2. The ratio of young fish to old – if this is decreasing it may indicate a decline in the water quality.</td>
</tr>
<tr>
<td>3. Trends in the amount of silting – an increase in silting may indicate poor vegetation.</td>
</tr>
<tr>
<td>4. Quantity of algae – if algae is increasing, nearby streams may be blocked.</td>
</tr>
<tr>
<td>5. Aerial photos – can indicate the status of the vegetation coverage, especially if multispectral capability is used. Amount of thriving versus dying vegetation can be determined.</td>
</tr>
<tr>
<td>6. Water quality of the aquifers – can be an indicator of the pollution level of the streams and lakes.</td>
</tr>
<tr>
<td>7. Visual inspections – provide a general qualitative means of judging the environment.</td>
</tr>
<tr>
<td>The more pristine an area the better the quality of the water.</td>
</tr>
</tbody>
</table>

5 ANALYSIS TECHNIQUES

5.1 Charts, Graphs, and Techniques

Charts and graphs are a good way to present measurement data. They are useful for displaying progress and performance over time. This section provides a brief discussion of 13 tools useful in the presentation and analysis of metrics\textsuperscript{21}.

5.1.1 Run Charts

Run Charts are used to illustrate trends or shifts in the average, to identify problems early in the life cycle as well as to perform analysis. This type of chart provides a clear picture of how smoothly the project is running; it does not indicate control. It provides a simple display of trends within an observation window over a period of time.

The example shown in Figure 5.1.1-1 shows a shift in the expected measurement, indicating a statistically unusual event. If the change is favorable, whatever caused that change should be made a permanent part of the process or system. If unfavorable, the cause should be identified and corrected or eliminated.

![Run Chart Showing Shift in Average Errors per Inspection](image)

Figure 5.1.1-1. Run Chart Showing Shift in Average Errors per Inspection

5.1.2 Control Charts

Control Charts are run charts with statistically determined upper and lower limits drawn on either side of the determined average. Control Charts allow a tolerance range and provide a means for analyzing project variables over time as well as identifying areas out of control or under control. Those measurements within the tolerance range are indications as well as those measurements out of tolerance. The tolerance limits are set when statistically significant sample sets have been collected.

\textsuperscript{21}Some examples have been adapted from The Memory Jogger, A Pocket Guide of Tools for Continuous Improvement, GOAL/QPC, 1985. For further information on measurement techniques, see Process Improvement Guide, Air Force Quality Institute, Maxwell Air Force Base, Alabama, 1993.
The Control Chart in Figure 5.1.2-1 illustrates Upper and Lower Specification Limits (USL and LSL) as well as Upper and Lower Control Limits (UCL and LCL). Specification limit refers to what an individual or an organization believes they need or what the customer requires. Control limit refers to what the process is capable of doing consistently. The curve on the left of the chart shows that the specification limits are much narrower than the control limits for the process. Resulting action is to improve the process or change the specifications.

![Control Chart Showing Process in Control, but Specification Limits Are Not Met](image)

Limits are calculated by running a process without adjustments or tweaking. Samples are taken at periodic times and plotted onto the graph. If any of the points fall above or below the line, something within the process is out of control and requires corrective action.

Fluctuation of the points within the bounds are usually caused by normal variations already built into the process (such as design decisions or choice of platform, etc.) and can only be affected by changing the platform or the design. Points outside of the bounds represent abnormal or special causes. Abnormal causes are usually related to people, errors, unplanned events, etc., and are not a part of the normal process.

### 5.1.3 Flow Charts

Flow charts are useful for analyzing and breaking down any process or task into smaller steps and identifying possible errors or problem areas. Use of this tool is especially good for visualizing a process thread for easier understanding and to identify areas where metrics would be appropriate, as illustrated in Figure 5.1.3-1.

The flow can also show the actual path versus the ideal path of a product or service. When trying to identify a problem, compare the actual steps versus the ideal steps to find the differences and where the problems will surface.
5.1.4 **Cause and Effect (Fishbone) Diagrams**

Cause and effect (or fishbone) diagrams are commonly used for problem identification; showing relationships between possible root causes and their effects. For each effect, there may be several major categories of causes. This tool helps to sort out and relate the various relationships to help facilitate finding the problem area(s).

Cause and effect diagrams are also useful for identifying areas where metrics would be beneficial. It is important to look to cure the cause and not the symptoms.

Figure 5.1.4-1 is an example of a cause and effect diagram showing relationships between root causes and effects. A cause and effect diagram is developed using the following steps:

1. **Specify the problem to analyze.** The effect can be positive (objective) or negative (problems). Place the problem’s title in a box on the right side of the diagram as shown in Figure 5.1.4-1.

---

2. List the major categories of factors influencing the effect you are studying. You can use the “4Ms” (methods/manpower/materials/machinery) as shown in Figure 5.1.4-1 or the “4Ps” (policies/procedures/people/plant) as your starting point.

3. Identify factors and subfactors. Ask yourself “Why?” or use brainstorming or mental imaging to generate ideas. Start with the major categories and work from there.

4. Identify significant factors. What factors appear repeatedly? List them. Then list the factors having a significant effect. (Your data can help you identify those.)

5. Prioritize your list of causes. Don’t confuse location with importance – a subfactor may be the root cause to all your problems. When you prioritize you may also discover new factors; then you will need to collect more data.

![Diagram of Causes and Effect]

Figure 5.1.4-1. Cause and Effect Diagram Showing Relationships Between Root Causes and Effect

5.1.5 Histograms

Histograms are often used to show a picture of process performance over a period of time. The Histogram displays data distribution and provides problem identification and analysis. Histograms may be used for a wide variety of data, such as labor hours predicted versus actual for various tasks, predicted labor hours over time, or SP/CRs per category of software problem or function. Histograms provide effective analysis of task durations for PERT networks to determine areas that require more detail. Data abnormalities and variations become evident such as in the example in Figure 5.1.5-1. This chart plots resource commitments vertically and time horizontally. Also shown are available resources and limits.
The following example was plotted from counts of tasks for each task duration up to 35 days (Figure 5.1.5-2). This was used to discover that most task durations on the PERT network being analyzed were much longer than the optimum 4 or 5 days, a risky situation when there is a tight schedule.

### Task Durations

This PERT has many long tasks: preferable to decompose each into several shorter tasks.

### 5.1.6 Pareto Diagrams

A Pareto diagram is a form of bar charting used to illustrate the relative contributions of a number of causes to an observed problem. As a metric display, it identifies the problems to be worked and is used as a lead-in for solving the problem. It can be used to identify the root cause of the problem, show the impact of the problem, and monitor the problem as measures are taken to bring the problem under control. It allows a view of all of the causes or conditions at once in order to choose a starting point. The height of the bar represents the number of instances of that condition or cause that were observed during the analysis of the problem. By convention, the cause contributing most to a problem is represented on the left, with other causes sorted in descending order of occurrence. A line graph shows cumulative contribution of the causes to the total problem. The example shown in Figure 5.1.6-1 charts primary causes for schedule impact or deviation versus number of tasks affected by each.
5.1.7 Scatter Diagrams

Scatter diagrams are typically used for problem analysis and can assist in identifying potential metrics. The scatter diagram illustrates relationships between two variables and testing of those relationships. When one variable’s changes affect the relating variable, this confirms that a relationship exists and indicates the strength of that relationship.

Direction and tightness of the clustering in Figure 5.1.7-1 show the strength of the relationship between variable A and variable B. The more the cluster resembles a straight line, the stronger the relationship between the variables. Scatter Charts are also useful to compare effort versus duration or size, time versus size, or faults versus size.
5.1.8  Check Sheets

Check Sheets are often used for counting and stratifying. This particular chart is appropriate to use for several metrics as a data collection and classification tool, but does not work well as a graphical display. The example in Figure 5.1.8-1 shows the number of defects per type over a series of design inspections.

Sample observations are gathered in order to detect patterns. This is a logical starting point in most problem solving cycles.

<table>
<thead>
<tr>
<th>Defect Type</th>
<th>1st Inspect</th>
<th>2nd Inspect</th>
<th>3rd Inspect</th>
<th>4th Inspect</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Wrong</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28</td>
</tr>
<tr>
<td>Major Missing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12</td>
</tr>
<tr>
<td>Major Extra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>33</td>
</tr>
<tr>
<td>Minor Wrong</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Minor Missing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
</tr>
<tr>
<td>Minor Extra</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Latent</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>18</td>
</tr>
</tbody>
</table>

|             | 39 | 31 | 54 | 23 | 147 |

*Figure 5.1.8-1. Check Sheet Showing Number and Type of Defects Found per Design Inspection*

5.1.9  X-t Plots

The X-t plots show activity over time, in this case achievement over time for selected milestones. A 70% to 92% range exists for projects installing the completed systems at site on time. Completion of system test ranges from 86% to 100% on all projects during this time period. This company invested in quality and process improvement programs during the middle of 1989. Their established improvement programs start to really pay off during calendar year 1991. The company’s achievements continue to scale upward even with the increase in the number of projects.
Network diagrams (also known as PERT charts), shown notionally in Figure 5.1.10-1, are a primary output of project management. This chart shows the relationships between tasks through logical formatting. Interdependencies are shown as well as resource data, completion percentages, actual start and stop dates, and estimates and durations. Using on-line tools, the PERT chart can be customized to reflect any number of variations. The information gathered while monitoring PERT diagrams is used as a basis for a number of metrics analyses. An example is shown in Figure 5.1.10-2.

Data on expected start and completion dates and actual start and completion dates for program tasks are collected and can be presented in both tabular and graphic form. The basic data can be analyzed in a number of ways, including comparison with plan and comparison to control limits among others.

Figures 5.1.10-3 and 5.1.10-4 show analyses of composite data collected over a 9-month period. Information collected over time represents a valuable resource for lessons learned and trend analysis.

Figure 5.1.9-1. X-t Plot Showing Achievement Results for Milestones over Seven-Year Period

5.1.10 Network Diagrams

Network diagrams (also known as PERT charts), shown notionally in Figure 5.1.10-1, are a primary output of project management. This chart shows the relationships between tasks through logical formatting. Interdependencies are shown as well as resource data, completion percentages, actual start and stop dates, and estimates and durations. Using on-line tools, the PERT chart can be customized to reflect any number of variations. The information gathered while monitoring PERT diagrams is used as a basis for a number of metrics analyses. An example is shown in Figure 5.1.10-2.

Data on expected start and completion dates and actual start and completion dates for program tasks are collected and can be presented in both tabular and graphic form. The basic data can be analyzed in a number of ways, including comparison with plan and comparison to control limits among others.

Figures 5.1.10-3 and 5.1.10-4 show analyses of composite data collected over a 9-month period. Information collected over time represents a valuable resource for lessons learned and trend analysis.
Critical Tasks

Noncritical Paths

Critical Path

Start

Stop

Task 1

Task 2

Task 3

Task 4

Task 5

Task 6

Noncritical Tasks

Figure 5.1.10-1. Network Diagram Showing Critical vs. Noncritical Paths and Tasks

<table>
<thead>
<tr>
<th>TITLE</th>
<th>Actual Start</th>
<th>Early Start</th>
<th>OVER DUE</th>
<th>Late %</th>
<th>Tot Exp Start</th>
<th>Early Finish</th>
<th>Late Finish</th>
<th>% MS Complete</th>
<th>MS Past Expected Finish</th>
<th>% Past Due</th>
<th>Total MS Due</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Milestones</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HW</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>33.33%</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>100.00%</td>
<td>0</td>
<td>0.00%</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 Third week. Subdividing ordering tasks to better estimate parts availability.</td>
</tr>
<tr>
<td>SW</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>25.00%</td>
<td>8</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>83.33%</td>
<td>1</td>
<td>16.67%</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6 HSA issues--second logical string established in network.</td>
</tr>
<tr>
<td>PMN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>8</td>
<td>0</td>
<td>3</td>
<td>27.27%</td>
<td>11</td>
<td>2</td>
<td>7</td>
<td>0</td>
<td>87.50%</td>
<td>1</td>
<td>12.50%</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

Figure 5.1.10-2. Tabular Depiction of Progress Data including Percentage Completion of Tasks Collected from Network
Figure 5.1.10-3. Graph of Milestone Performance Over Time Collected from IPT Meetings

Figure 5.1.10-4. P-Bar Control Charts Used to Detect Performance Outside Control Limits

5.1.11 Gantt Charts

The example Gantt chart in Figure 5.11-1 provides another way of looking at task durations or time scales. The length of each bar represents the relative duration of the task it represents. The bar also shows the start and end dates of each task against the time scale as well as the critical task (path). A stair-step pattern often indicates a critical path, as the end of each task matches the required start date for the next task. Overlaps in the horizontal bars may indicate negative float on a critical path. Gaps between the end of one bar and the beginning of another can be interpreted as positive float, an amount of “breathing room” in the schedule. Gantt charts may be annotated to indicate delayed tasks and actual progress to date.
5.1.12 Line Graphs

Line graphs are often used to display cost and resource information. The cost graph in Figure 5.1.12-1 displays that the budgeted cost of work scheduled (BCWS) to be performed for the project is lower than the budgeted cost of work actually performed (BCWP) during implementation. However, the actual cost of work performed (ACWP) is higher, including unplanned expenses incurred during implementation.

![Figure 5.1.12-1. Cost Graph Showing BCWS Lower Than BCWP and Higher Than ACWP](成本图示图)

5.1.13 Pie Charts

Pie charts provide a simple way to display percentages. The example in Figure 5.1.13-1 illustrates the results of functional testing and the percentage of Trouble Reports written. This chart, along with a histogram (see paragraph 5.1.5 for discussion of histogram) to correlate the number of tests conducted per thread will
provide information about the number of bugs being found. Action on the tester’s part would be to either exercise the other test scenarios more, or take a closer look at the problem areas.

![Pie Chart Showing Percentage of Trouble Reports per Test Thread]

### 5.2 Spreadsheets

Spreadsheets can be used to establish data links between external software applications to other tools, such as those used for project management for updating and gathering data. For example, an external financial or timekeeping application spreadsheet, when linked to project management tools, will update actual progress data automatically. Or, in a specific case of multiple inputs to one spreadsheet, functions such as publish and subscribe support multiple inputs and automatically port these inputs to a master spreadsheet. This is particularly useful when there are several metrics analysts or several engineers responsible for gathering raw metrics data. Each individual has access to his/her own spreadsheet, but the data is then transferred (transparently to the user) to a master spreadsheet or file.

Other tools such as desktop automation applications can be linked to spreadsheets and used for metrics data gathering and analysis. These require a little more effort to set up but provide a friendlier user interface, customized for each type of user.

### 5.3 Databases

Databases are the preferred way to set up a historical company profile for each project. Storing metric data in a database environment provides easy access and search routines for trend analysis, estimations, and predictions.

Since measurement is an iterative process, the more we collect measures within the context of the environment, the more we learn about what needs to be collected. As we transition through the iterations of measurement, our main objectives are to be able to quantify the benefits and to improve return on the investment made. A good way to accomplish these objectives is to set up a permanent database and feed in data from every project involved in data collection. In this way, the data becomes a resource for the whole division or organization, resulting in more sophisticated data analysis and reporting capabilities.
6 DEVELOPING METRICS: AN AEROSPACE EXAMPLE

6.1 General

Chapters 2 and 3 described two techniques for development of metrics sets, QFD and GQM. This chapter deals with the “nuts and bolts” of developing metrics using an aerospace example. It describes the concept of a physical “Metric Package,” and provides a step-by-step process for developing metrics. This approach, using the system operational definition to guide development of a metrics approach, can also be used in a commercial context.

6.2 The Metric Package

A metric consists of three basic elements: (1) the operational definition, (2) the actual measurement and recording of data, and (3) the metric presentation. Together these elements are called the “metric package,” illustrated in Figure 6.2-1. The operational definition is the precise explanation of the process being measured. The measurement and data collection is the translation of data from the process into understandable and useful information. The metric presentation is the metric’s communication link to the product team and process owner.

![Figure 6.2-1. The Metric Package](image)

The first element of the metric package is the operational definition. (The template shown in Chapter 3 is an example.) According to some “there is nothing more important for the transaction of business than the use of operational definitions.” The operational definition is the who, what, where, and how of the metric. It is customer oriented and accepted. It must be made over time and is the key to internal communication for

---

23 The metric development process and examples in this chapter were extracted from *The Metrics Handbook*, HQ AFSC/FMC, Andrews AFB, D.C., August 1991, 1st Edition
process understanding. The detail required will vary from metric to metric, but should consist of at least the following elements:

1. An unambiguous description of the metric
2. Population that the metric will include
3. Frequency of measurement
4. Source of data
5. Equations required in doing the measurement
6. Precise definition of key terms
7. Description of the graphic presentation that will eventually be used to display the data
8. Customer of the metric
9. Accountable process owner
10. Desired outcome expressed in terms of positive or negative trend (not a numerical goal)
11. Link between the process being measured, your organization’s strategic plan, and the goals.

The second element is the actual measurement of the process. It is the collecting and recording of data. This step is the heart of the metrics process and serves as the translation of data from the process being measured to meaningful information that will be the basis of process improvement activities.

The third element of the metrics package, the metric presentation, has two parts – the metric descriptor and the graphic presentation of the data. The descriptor and presentation are actually planned when developing the operational definition, before most of the data is collected. The amount of detail required in the descriptor will vary with the purpose of the presentation. It should contain enough information to clearly communicate the information contained in the second part of the presentation. The second part of the metric presentation, the chart, is often mistakenly referred to as the metric. Together the descriptor and the chart should provide enough information to clearly communicate the metric.

### 6.3 Steps to Metric Development

This section provides the steps for formulating the metric package and measuring associated process improvement. Keep in mind that these steps may require tailoring before they can be applied to your organization. The steps are pictured with a combined Air Force/Industry example in Figure 6.3-1.

**Step I. Identify Your Purpose**

It is important to first align your purpose with your organization's mission, vision, goals, and objectives. These should be inextricably linked to meeting customer needs and serve as a foundation for accomplishing and sustaining continuous, measurable improvement.

**Step II. Develop Your Operational Definition, Starting with Your Customer**

Define the who, what, when, why, and how of this metric in sufficient detail to permit consistent, repeatable, and valid measurement to take place. The operation definition starts with an understanding of your customer’s expectations. You then “operationalize” the expectation(s) by defining characteristic(s) of the product, service, or process that are internally measurable and which if improved, would better satisfy your customers’ expectations. This is actually an iterative process involving Steps II through VII. This is the first element of your metric package.

**Step III. Identify and Examine Existing Measurement Systems**
Once the link to objectives and goals has been established, it is essential to determine if existing metrics or other measurement systems exist that satisfy your requirements. Don’t “reinvent the wheel.” Use existing process measurements when they exist. (See Chapter 4, Metrics Framework.)
Step IV. Generate New Metrics if Existing Metrics are Inadequate
Most measurements used in the past were not process oriented. They were results indicators related to final outputs, products, or services for external customers. With metrics, the focus includes the process input in making these final outputs. These upstream process measures drive the final outcome and are the key to making process improvements. When process performance is monitored and improved, the quality of the products and service improves.

Step V. Rate Your Metric Against the “Eight Attributes of a Good Metric”
Refer to the attributes listed in Chapter 2. If you feel your metric sufficiently satisfies these criteria for a good metric, go to Step VI. If not, return to Step II and correct the deficiencies.

Step VI. Select Appropriate Measurement Tools
Select the proper tool for analyzing and displaying your data. The tools discussed in Chapter 5 are the most common. However, other statistical and nonstatistical tools may be more appropriate for your application.

Step VII. Baseline Your Process
Start acquiring metric data. This serves as a baseline for determining the capability of your process. Ask if the data accumulated over time adequately measures the important characteristics of your process. If the answer is uncertain, examine other possibilities. When metrics are changed, remember to coordinate it with the team(s) and your customer again.

Step VIII. Collect and Analyze Metrics Data Over Time
Continue aggregating metric data over time. Examine trends. Special and/or common cause effects on the data should be investigated and assigned. Compare the data to interim performance levels. This is the second element of your metric package.

Step IX. Finalize the Metric Presentation
Based on the results of the previous steps, you are finally ready to present the metric externally. The descriptor will provide enough information to communicate the appropriate details of the metric to your customer. The appropriate level of detail should be determined by discussion with the customer. This information should be an abbreviation of the key elements of the operational definition. The graphic presentation clearly and concisely communicates how you are performing. This is the third element of your metric package.

Step X. Initiate Process Improvement Activities
Initiate process improvement activities in conjunction with the key process owners. Once improvements have been implemented, the process above may start over or it may pick up again at almost any step. Remember that metrics are just a means to an end! That end is continuous process improvement.
### Metric Development Process

<table>
<thead>
<tr>
<th>I. Identify your purpose or objective</th>
<th>I. Improved communications with our users</th>
</tr>
</thead>
<tbody>
<tr>
<td>II. Develop and coordinate your metric operational definition starting with your customer</td>
<td>II. At least biannual communications with users and respective program offices using mutually coordinated &amp; approved “Customer Satisfaction Assessment Reports”</td>
</tr>
<tr>
<td>III. Identify existing metrics</td>
<td>III. No existing corporate metrics in place</td>
</tr>
<tr>
<td>IV. Generate new metrics if required</td>
<td>IV. Customer satisfaction indices using scales 1 (very dissatisfied) to 6 (very satisfied) in 3 areas</td>
</tr>
<tr>
<td>V. Rate your metric against attributes</td>
<td>V. - Program execution</td>
</tr>
<tr>
<td></td>
<td>- Program cost</td>
</tr>
<tr>
<td></td>
<td>- Product acceptability</td>
</tr>
<tr>
<td></td>
<td>- Working relationships</td>
</tr>
<tr>
<td></td>
<td>- Optional topics</td>
</tr>
</tbody>
</table>

*Continued on next sheet*

*Continued on next sheet*

---

**Figure 6.3-1. Metric Development Process (Sheet 1 of 4)**
Figure 6.3-1. Metric Development Process (Sheet 2 of 4)
Figure 6.3-1. Metric Development Process (Sheet 3 of 4)
Figure 6.3-1. Metric Development Process (Sheet 4 of 4)
7 METRICS TAILORING AND MANAGEMENT PRIORITIES

Organizations add or modify metrics and measures to meet the needs of specific customers, programs, and projects while preserving the integrity and general principles of the measurement process. This tailoring of the metrics set is an iterative process, and it requires management and team involvement to reach consensus. Figure 7-1 illustrates the steps involved in tailoring the metrics of a program.

Based on internal and external goals and requirements, the example set can be tailored to either add metrics, modify existing metrics (to better represent the intent of the program’s goals), or delete metrics when programs are small and do not have the resources or need for this amount of measurement. For example, a project that does not require software would just use metrics for counting software lines of code.

As with goal setting, tailoring should be done cooperatively by participants from various organizations or disciplines. Keep in mind the different considerations that the various levels of participants bring to the tailoring activity. A brief description addressing some of these considerations is given in the following paragraphs.
7.1 Executive Considerations

Executive management wants to see the broadest level of impact to ensure that the decisions are in line with customer needs, corporate directives, corporate goals, and financial objectives.

7.2 Project Management Considerations

Project management’s foremost concern is with how the metrics will affect the products, the benefits to the customer and the project, and the cost of the metrics effort.

7.3 Integrated Product Team Considerations

Team members will need training and management perspective to guide their selection of technical performance, progress, and process metrics appropriate to the specific product or system.

7.4 Tailoring Example – Mustang Program

Mustang is a project, executed by one of the member companies, designed around an integrated product team (IPT) approach. A key part of the project plan was to build a metrics program and use that set of metrics to help measure the effectiveness of an IPT approach.

The first Mustang metrics tailoring meeting was convened by the Hardware Laboratory Director and the program's Chief System Engineer. Nine engineers representing various disciplines were in attendance. At the first meeting, the members of the team were empowered with the responsibility to define and implement a metrics program on Mustang. Also, during the first meeting, the Metrician presented background information on the Core Metrics Sets and the Metrics Superset. In addition, a mini-training briefing was provided to the team, discussing purpose and benefits of measurement. (Approximate meeting time = 1.5 hours.)

A second meeting was held later that day with the same team members. Goals of the program were discussed and an initial cut of the example metrics set was made by the team. (Approximate meeting time = 3.0 hours.)

Iterative meetings were held to refine the remaining set of metrics. New metrics were added, based on the customer's requirements, technical performance measures, and the program’s goals. Existing metrics were modified in compliance with the intent of the goals. Other metrics were deleted due to the size of the program and resource constraints. Periodic metrics collection points were redefined due to the variance in the customer's required reviews relative to the program schedule. The results from these meetings were compiled and cleaned up by one of the metricians. (Approximate time for cleanup = 4.0 hours)

A sanity check was conducted to ensure that the finalized set of Mustang metrics was achievable based on the available resources, currently available tools, and those tools to be acquired. The Mustang metrics set was refined one last time. (Approximate time = 3.0 hours.)

Results were presented and signed off by the Chief System Engineer.

Using the selected metrics, the metrics job was scoped (in labor hours), taking into consideration the available resources and tools. At the completion of the design phase, an additional set of software and hardware development and quality metrics was added to the Mustang metrics set.
Note: Tailoring the Core Metrics set to Mustang took approximately 75 total labor hours, including the participation of the product IPT leaders.
8 MANAGING BY THE DATA

“Managing by the data” begins with collecting data about a specific topic, area, or subject matter that is relevant, appropriate in amount, and timely. Through analysis, this data is turned into “meaningful information” that provides both insight and knowledge about the area, topic, or subject matter. This knowledge allows the manager to take appropriate action(s) based on the resulting analysis, thus reducing uncertainty and providing a clearly defined understanding of the subject matter, topic, or area.

The following diagrams illustrate various kinds of metrics that relate to management goals derived through the goal/question/metric approach, described earlier in this guidebook. These metrics provide a broad range of answers to a balanced set of questions without creating excessive costs in their collection and analysis. Goals and questions related to the measures will be discussed with each one.

8.1 Customer Satisfaction

One of the first and foremost goals of any company is to maximize customer satisfaction. The graph in Figure 8.1-1 captures the customer’s viewpoint in the areas of overall satisfaction, functionality and usability of the product, and accuracy of supporting documentation. Collection of data for this type of metric can be accomplished through sophisticated means such as Usability Labs and real-time monitoring, or collection can be accomplished by simply using feedback mechanisms such as on-line surveys or questionnaires, where the customer can enter data periodically throughout the development cycle. If the development is product oriented versus systems oriented, then add problem reports and defect inspections to the metrics for customer satisfaction.

---

This graph in Figure 8.1-1 answers four questions related to the goal: (1) Is the customer satisfied with the end product or system? (2) Is the supporting documentation accurate and useful? (3) Is the system user friendly? and (4) Does the system or product provide functionality in accordance with customer requirements?

### 8.2 Schedules and Costs

One of the primary goals for management is to improve upon schedule accuracy. To do so, takes into consideration time and how accurate the original estimates are, as well as the interrelationship of project estimates and program milestones. Capturing major milestone schedules is not enough. All milestones should be monitored and tracked on all programs, as shown in Figure 8.2-1 for detailed milestones. These estimates and actuals feed into a larger picture of project progress rate, illustrated by Figure 8.2-2.
Each point on the graph in Figure 8.2-2 represents the monthly project progress rate (ratio of completed and projected milestones to estimated milestones) averaged for the past several months\(^{25}\). The ideal progress rate is 1 when milestones are being accomplished according to the original schedules. This graph uses a moving average to smooth point-to-point swings. It answers two questions related to the goal: (1) How accurate are our schedule estimations? and (2) Are our schedules improving?

---

Figure 8.2-3 illustrates a sample graph of life cycle costs.

**Budgeted cost of work scheduled** (BCWS) is the sum of the budgets for all work breakdown packages, the level of effort, and apportioned effort scheduled to be accomplished within a given period of time.

**Budgeted cost of work performed** (BCWP) is the sum of the budgets for completed work breakdown packages and completed portions of open work breakdown packages, plus the applicable portions of the budgets for level of effort and apportioned effort.

**Actual cost of work performed** (ACWP) is the cost actually incurred in accomplishing the work performed within the given time period.

Cost and schedule variances for the example shown in Figure 8.2-3 are illustrated in Figure 8.2-4. It is over cost and behind schedule. Variances of zero mean that the planned budget and schedule are being met.

The difference between planned and actual cost, \( cost\ variance = BCWP - ACWP \).

The difference between the amount of work planned to be completed and work actually completed, \( schedule\ variance = BCWP - BCWS \).

The techniques described above are often referred to as “Earned Value” (EV). Although the term EV is derived from the government/aerospace program environment, the techniques themselves are applicable to all product development environments.\(^{26}\)

8.3 Productivity

Primary costs are often in engineering months and calendar months; however, productivity measurement is a sensitive area. Therefore, it is best to use this measure in cooperation with process improvement efforts that include regular inspections on all programs as part of the life cycle process, failure analysis, and project plans.

The goal here is to improve life cycle productivity. Measurement of productivity can be anything from pages of documentation developed to SLOC, as illustrated in Figure 8.3-1. The question answered by this metric is What progress are we making toward improving our life cycle processes for development of products? The examples described below were taken from software product development programs.

Analysis of the RIZA and RSTV projects:

Riza Project: 5 people

37 project months x 5 people = 185 man-months (mm)

218K SLOC / 185 mm = 1178 SLOC per mm / 5 people = 236 loc developed per person per mm.

The normal productivity rate for developing new Ada code is 1.2 hours per line of code. This is based on 151 hours per month / average loc per person [126] = 1.2 productivity rate. The Riza project's productivity rate is 0.6 hours per line of code, one-half the average and close to twice the average code amount.
Success of this project might be attributed to a combination of project factors such as expert Ada programmers; a well-defined process; clear, stable requirements and design; and good communications due to a small, co-located team. These types of project attributes and others are to be taken into consideration as part of the productivity equation and can be gathered from project parametric modeling reports.

<table>
<thead>
<tr>
<th>Projects / # People</th>
<th>Project Size (source lines of code)</th>
<th>Cost (man-months)</th>
<th>Project Duration (months)</th>
<th>Productivity SLOC (man-months)</th>
<th>SLOC Developed per person per mm</th>
<th>Hours per SLOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIZA / 5</td>
<td>218,000</td>
<td>185</td>
<td>37</td>
<td>1178</td>
<td>235.6</td>
<td>0.6</td>
</tr>
<tr>
<td>PTR / 7</td>
<td>110,000</td>
<td>182</td>
<td>26</td>
<td>604</td>
<td>86.3</td>
<td>1.7</td>
</tr>
<tr>
<td>CBBT / 5</td>
<td>39,000</td>
<td>60</td>
<td>12</td>
<td>650</td>
<td>130.0</td>
<td>1.2</td>
</tr>
<tr>
<td>RSTV / 10</td>
<td>485,000</td>
<td>450</td>
<td>45</td>
<td>1078</td>
<td>107.8</td>
<td>1.4</td>
</tr>
</tbody>
</table>

![Figure 8.3-1. Collection of Trend Data Showing Productivity Rates](image)

**RSTV Project: 10 people**

45 project months x 10 people = 450 man-months (mm)

485K SLOC / 450 mm = 1078 SLOC per mm / 10 people = 107.8 SLOC developed per person per mm.

The RSTV project’s productivity rate is 1.4 hours per line of code – less than the average. Compared to the Riza project, the RSTV project has twice the code and twice the number of people; however, the productivity rate is less than half that of the Riza project. Any number of conditions can change the productivity average such as new Ada programmers, distributed locations, unstable requirements, changing design, and so on.

Since the various projects’ productivity rates are within the range of 0.6 to 1.7, future estimations of lines of code count on similar Ada projects can be calculated, with confidence, at 1.2 hours per line of code or less (average of the range), based on trend data.

*(Note: Assumptions: $65 labor rate/hr x 151 hr/mo = monthly wage x man-months = cost in $$; all new Ada development, full documentation). Metrics data should be captured in a database schema over time to show productivity rates and costs for future estimations.*
8.4 Problem/Change Reports

Problem reports are also known as EDNs, PARs, or [System, Software, Hardware, or Test] Problem/Change Reports (SP/CRs, HP/CRs, or TP/CRs). Problem/Change reports are generated by anyone uncovering desired modification, changes, errors, or discrepancies in configuration-controlled components or documents.

![Graph of Problem/Change Reports](image)

Figure 8.4-1. Problem/Change Reports – Program

Problem/Change Reports are categorized by priority (such as immediate, urgent, routine) and by problem type. The most critical category of Problem/Change Reports should be closed as rapidly as possible.

The following diagram plots out the following measures at a program level.

- Number of open Problem/Change Reports
- Number of emergency (or critical) Problem/Change Reports
- Number of new Problem/Change Reports

Smaller diagrams in Figure 8.4-2 show the number of open Problem/Change Reports on all configuration items (HWCI or CSCI) with the total shown in the top right report.
8.4.1 Design/Hardware Problem Metrics

Design and Hardware Problem metrics reports, when produced regularly, reveal engineering documentation and hardware error trends and problem areas in four categories of design and manufacturing: (1) pre-release electronics, (2) post-release electronics, (3) pre-release mechanical, and (4) post-release mechanical as shown in Figure 8.4.1-1. The report provides a basis for investigation/analysis and actions to reduce errors in these areas, thereby reducing cost and cycle time. Pre-release errors are identified and categorized as dimensioning, part number, note content, depiction, etc. and are entered in a database for each program by part number. Post-release errors are extracted from engineering change orders or drawing revisions and are categorized by part number and program. Upper control limits are established and corrective action is implemented for recurring problem areas.
Figure 8.4.1-1. Mechanical Pre-Release Documentation Data (by Error Category)
8.5 Process Improvement

Another important area for measurement is process improvement. Process improvement can be monitored and tracked in many ways, but no other method is so well established as the Software Engineering Institute's (SEI’s) Software Capability Maturity Model (CMM), the developer of the process improvement approach shown below. The majority of this section uses the software quality equations as an illustration of possible process measurements techniques, potentially applicable for broader application when the Systems Engineering and Integrated Product Development Capability models, currently in development, are completed. Interestingly, the engineering and business theory communities both focus on using process improvement as the key mechanism for obtaining reductions in product design and development cycle time.

The benefits of reduced cycle time are realized not only in schedule, but also in effort/cost, and reliability (few defects (or Mean Time to Defect (MTTD)). A software equation used by Quantitative Software Management, Inc. (QSM) validates the benefits of process improvement. Conceptually, the equation

\[
\text{Quantity of Function} = \text{Process Productivity} \times \text{Effort} \times \text{Schedule}
\]

which says that the product of the time and effort coupled with the process productivity of the organization determines how much function can be delivered. Taking into consideration that very strong nonlinearities exist in software behavior, the calculations from the software equation discloses these nonlinearities and how management can exploit them, shown as

\[
SLOC = \text{Process Productivity Parameter} \times (\text{Effort}/B)^{1/3} \times \text{Time}^{4/3}
\]

where

- The Process Productivity Parameter (PPP) is the development process efficiency in the performing organization. This is determined from historical data.
- SLOC is Source Lines of Code (other measures can be used)
- Effort is the person-years of effort required including all types of labor used on the project.


• B is a factor that provides for specialized skills for integration, testing, documentation, and management as the size of the system increases. It can be a complexity adjustment factor for size or remain a constant as in these examples.

• Time is the elapsed calendar time in years from the start of detailed design until the product is ready to be operationally installed (frequently this is a 95% reliability).

If the equation is rearranged to isolate effort and multiplied by a fully burdened labor rate, then we obtain software cost equation

\[ SW \text{ Cost} = \frac{\text{SLOC}}{\text{PPP}} \cdot \left(\frac{1}{\text{Schedule}^4}\right) \cdot (B \cdot \text{LaborRate}) \]

This equation provides enormous economic leverage to be exploited. If size is reduced through reuse, then the cost is reduced by the cube of the size reduction. If process productivity is increased through investment in tools, training, and discipline, then the cost goes down by the cube of the increase in Process Productivity Parameter. If a smaller team is used and the schedule is modestly lengthened to accommodate this, then the cost is reduced by a fourth power of the schedule increase. On any given project, sometimes two or three of these situations can be made to happen.

In Figure 8.5-1, benefits from process improvement, using the Process Productivity Parameter (PPP) are demonstrated. When using the PPP in the software equation, it is equivalent to moving up the SEI maturity scale. A linear correlation is established between the SEI scale and its Productivity Index, which linearizes and represents the PP term in the Software Equation.

![Figure 8.5-1. Benefits of Process Maturity – Level I to Level III](image)

L.H. Putnam, “The Value of Process Improvement: Does It Pay Off?” 4th International Conference on Applications of Software Measure (ASM ’93). The Productivity Index (PI) is a simple linear number that is used to transform the non-linear values of the PPP (calculated in the software equation) into an easy to interpret linear scale. The scale ranges from 1 to 40. All software projects measured so far fit within this range.
8.6 Performance

This metric provides a monthly report of cost and schedule variance totals for any organization.

SBCWS = Budgeted cost of work scheduled, summed over all specific (SW sys, test, HW, etc.) engineering organization accounts.

SBCWP = Budgeted cost of worked performed, summed over all specific engineering organization accounts.

SACWP = Actual cost of worked performed, summed over all specific engineering accounts.

Schedule Variance = SBCWS/SBCWP

Cost Variance = SBCWP/SACWP

The objective of this metric is to drive the schedule variance to 1.0 and to maintain cost variance greater than or equal to 1.0. Figure 8.6-1 is an example.
8.7 Technology Infusion

New technologies are introduced into organizations in two ways:

1. **On the Job** – This is the activity that requires new technology be used to solve technical problems, redesign a system, anticipate requirements, etc. It occurs in product development, in custom contract work, on special projects, and IR&D. The metric is the number of hours per month per person that are devoted to working with new technologies. It is captured from the monthly labor hours reports. Cost accounts are labeled as either (1) new technology or (2) old or non-technical.

2. **Training** – hours of new technology from the classroom per month per engineer (e.g., internal and external classes, Tech Tuesdays, funded and unfunded, etc.).

The objective is to track trends and maintain a critical level of technology infusion. Figure 8.7-1 shows an example of this type of tracking.

![Technology Infusion Graph](image-url)

*Figure 8.7-1. Tracking of New Technologies*
8.8 Software Reuse

Software reuse, as shown in Figure 8.8-1, is the percentage of code bid, for all software programs, as unmodified reuse and percentage of code developed with unmodified reuse. Unmodified reuse includes:

1. Internally developed code.
2. External code that is internally maintained.

Commercial-off-the-shelf (COTS) and government-off-the-shelf (GOTS) code is not included.

![Software Reuse Tracking Graph]

The objective is to reduce time to market, system development, and maintenance costs by increasing the amount of software reuse.

8.9 Key Process Area Satisfaction Profile

The metric for Key Process Area Satisfaction Profile is based on assessing and examining project satisfaction of the Goals of each Key Process area of the SEI’s Capability Maturity Model (CMM). There are 18 Key Process Areas (KPAs) in the CMM, shown in Figure 8.9-1. Fifty-two goals are associated with these KPAs. The degree of satisfaction of each goal is determined by application of a variant of the Software Capability Evaluation techniques taught by the SEI. For each project, an appraisal can be made as to the degree of satisfaction for each of the 52. Each of the goals can be
examined with respect to elements of commitment to perform, ability to perform, the actual activities performed, measurement and analysis of the activities, and verification of the activities. Weights are assigned to these elements to map each project to the scale of Some, Fair, Meets, and Fully Meets with the following meanings:

1. **Some Evidence**: There is observable evidence of goal satisfaction, even though the evidence may be anecdotal and have a limited documentation trail.

2. **Good Evidence**: Most of the project personnel agree that the practice is in place and being followed; the documentary evidence is known at least to the project leadership.

3. **Meets Need**: The perceived needs of the company and/or the specific project are met by the practices that are in place; the predominate view on the project is that the practice is followed sufficiently.

4. **Fully Satisfies**: The practices and documentary evidence of the practices meet all the expectations of the SEI CMM. (As example data, 13 features are examined for full satisfaction and described later in this section.)

The degree of satisfaction is determined by a trained team of experienced software professionals, using a directed interview technique and document review. Since the 52 subprocess areas are too broad to characterize with a single yes/no answer, the teams examine 13 features described in subparagraphs a. through m. below. These are an elaboration of the Five Common Features described in the CMM. In order for a particular subprocess area to be judged “Fully Satisfied,” these features are expected to be present with respect to the subprocess area:

---

**Figure 8.9-1. Software CMM KPAs**

<table>
<thead>
<tr>
<th>Repeatabler Level KPAs (Level 2)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Mgt.</td>
<td>RM</td>
</tr>
<tr>
<td>S/W Project Planning</td>
<td>PP</td>
</tr>
<tr>
<td>Software Project Tracking and Oversight</td>
<td>PT</td>
</tr>
<tr>
<td>S/W Subcontract Mgt.</td>
<td>SM</td>
</tr>
<tr>
<td>S/W Quality Assurance</td>
<td>QA</td>
</tr>
<tr>
<td>S/W Configuration Mgt.</td>
<td>CM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Defined Level KPAs (Level 3)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Org. Process Focus</td>
<td>PF</td>
</tr>
<tr>
<td>Org. Process Definition</td>
<td>PD</td>
</tr>
<tr>
<td>Training Program</td>
<td>TP</td>
</tr>
<tr>
<td>Integrated S/W Mgt.</td>
<td>IM</td>
</tr>
<tr>
<td>S/W Prod. Engineering</td>
<td>PE</td>
</tr>
<tr>
<td>Intergroup Coordination</td>
<td>IC</td>
</tr>
<tr>
<td>Peer Reviews</td>
<td>PR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Managed Level KPAs (Level 4)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantitative Process Management</td>
<td>PM</td>
</tr>
<tr>
<td>Software Quality Mgt.</td>
<td>QM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Optimizing Level KPAs (Level 5)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Defect Prevention</td>
<td>DP</td>
</tr>
<tr>
<td>Technology Change Mgt.</td>
<td>TC</td>
</tr>
<tr>
<td>Process Change Mgt.</td>
<td>PC</td>
</tr>
</tbody>
</table>
1. **Commitment to Perform** (the actions taken to ensure that the subprocess area is implemented and will endure)
   a. *Leadership* – the assignment of responsibility and the presence of sponsorship
   b. *Organizational policies* – there are written policies governing the subprocess area

2. **Ability to Perform** (the preconditions to implement the subprocess area competently exist in the project or organization)
   c. *Resources* – the adequacy of resources (e.g., staff, funds, facilities, tools)
   d. *Organizational structures* – the organizational structure provides support for the process activities (e.g., job descriptions, defined relationships between entities on the organization chart)
   e. *Training* – availability of pertinent training and orientation, and its timeliness for the people who carry out the activities in the implementation of the subprocess area (e.g., curriculum content, training schedule, records)

3. **Activities Performed** (the roles and procedures necessary for implementation of the processes)
   f. *Plans and procedures* – plans and procedures exist and are prepared according to a documented procedure
   g. *Work performed* – the objective evidence of the use of plans, procedures, and standards in the work done by the organization (i.e., the track record and "paper or electronic trail")
   h. *Tracking* – how the work is tracked and how problems are identified
   i. *Corrective actions* – the identification and resolution of problems

4. **Measurement and Analysis** (the determination of the status and effectiveness of the activities)
   j. *Measure process* – the measurements of activities performed (e.g., resources consumed, problems encountered, work product characteristics, and status of activities)
   k. *Analyze measurements* – the analysis and use of measurements taken

5. **Verifying Implementation** (the actions that ensure compliance to established practice)
   l. *Reviews* – management reviews
   m. *Audits* – there are audits undertaken of activities and work products.

As projects are examined, their goal satisfaction results are added to a database for the organization as a whole. The results should be presented to management, and to the projects, as a chart of the range of scores over the set of projects. Figure 8.9-2 shows how example data can be presented for the company as a whole, with the square dot indicating the average for the projects examined. When this chart is shown to the individual projects as feedback on the assessment, the score of each project for each KPA was added with a contrasting symbol. In this way both the management and the projects are able to see the KPAs that are being done well and the ones on which the company/project(s) need to focus improvement efforts. Although this same visualization technique can be used for all five levels of the CMM, this particular example assessment does not examine the Level 4 and Level 5 KPAs.
8.10 Technical Stability/Requirements Volatility

The stability of requirements is a major factor directly affecting the technical stability of a project. Requirements volatility is the extent to which the requirements of the product or system are undefined, unclear, or changing from the time of initiation of the project. This lack of definition, clarity, or stability is manifested principally in incomplete and contradictory technical requirements and sometimes in insufficient human resources. Insufficient schedule and budget can also be a contributor to technical instability, though not covered in this section. The terminology used in the following paragraphs is familiar to both product-oriented environments in the commercial sector as well as government projects.

**Definition**

*System and Allocated Requirements*

The following metrics are an example set to be collected for each large product or system component/subcomponent, also referred to as a CSCI, CSC or CSU, and the project. Associated graphs should be plotted for each large component (CSCI) or functionality area within a product or system and for the project as a whole.

a. Number of System Specification (SS) or Product Specification (PS) requirements
b. Number of SS requirements in the specifications for integrated product areas [such as an Software Requirements Specification (SRS) or a Hardware Requirements Specification (HRS)], as well as an Interface Requirements Specifications (IRSs) (those requirements connecting large functional components or objects within a product or a system) for integrated product areas.
c. Number of SRS and IRS requirements
d. Number of HRS and IRS requirements
e. Number of SRS and IRS requirements not traceable to the SS
f. Number of HRS and IRS requirements not traceable to the SS
g. etc. ...

**Software Requirements Discrepancies**
Many disciplines are involved in the development of a system or product. The following illustration provides a software example. The reviews referred to by “Review” in this section are defined below, describing familiar terms easily translated to those environments most familiar to the reader.

SSR : Software Specification Review  
PDR : Preliminary Design Review  
CDR : Critical Design Review  
TRR : Test Readiness Review

Each of the following metrics should be collected for each CSU, CSC, CSCI and Project. Associated graphs should be plotted for each CSCI as well as the project as a whole. Figure 8.10-1 shows an example of the graphs provided for this metric.

a. Number of Requirements Discrepancies per Review  
b. Number of Open Requirements Discrepancies  
c. Number of Closed Requirements Discrepancies
**Engineering Change Proposal (ECP) Statistics**

ECPs can represent large changes to the functionality or requirements of the system or product. These types of changes are often out of scope and greatly affect the schedule and cost of the project. Each of the following metrics should be collected for the Project as a whole as well as any associated graphs. A software example is again provided since SLOC is a term known across environments. Figure 8.10-2 shows an example of the graphs provided for this metric.

- a. Number of ECPs due to Customer
- b. Number of CSUs affected by ECPs due to Customer
- c. Number of SLOC affected by ECPs due to Customer
- d. Number of ECPs due to Contractor
- e. Number of CSUs affected by ECPs due to Contractor
- f. Number of SLOC affected by ECPs due to Contractor

![Source of ECPs - Cumulative](image)

**Figure 8.10-2. Source of ECPs – Cumulative**

Listed below are the different stages ECPs go through. These stages show the status of the ECP. When “Status” is used in the following metric it refers to this definition. It is a suggested way to track the status of ECPs with regard to those stages.

- Proposed
- Opened
- Approved
- Incorporated

The following metric should be collected for each CSCI and the project as a whole, as well as any associated graphs.

- g. Number of ECPs per their Status

**Requirements Changes**

The requirements changes metrics will be valuable in assessing the stability of the project. The specific categories under which the different requirements changes fall will also help in determining the severity of the
change to the project. This is an example set of Requirements Categories to track for requirements changes. This is the definition of “Requirements Category” when used in this section and throughout the document.

- Security
- System Configuration
- Interface
- Functional
- Performance
- Usability
- Operating Environment
- Reliability
- Maintainability
- Other Quality Requirements

Each of the following metrics should be collected for each CSCI and project, as well as any associated graphs. Figure 8.10-3 shows an example of the graphs provided for this metric.

a. Number of Requirements Changes
b. Number of Requirements Changes per Requirements Category

![Cumulative Requirements Changes](image)

**Figure 8.10-3. Cumulative Requirements Changes**

**Staffing Profiles**

Each of the following metrics should be collected for each CSCI and project. Associated graphs should be plotted for each CSCI. Figure 8.10-4 shows an example of the graphs provided for this metric.

a. Estimated Staffing Profile per Labor Category
b. Actual Staffing Profile per Labor Category
c. Unplanned Staff Losses per Labor Category
d. Unplanned Staff Gains per Labor Category
116

Figure 8.10-4. Project Staffing

Analysis

1. Technical stability of the project is important because stability is often the prime determinant of whether or not the project will be completed within the required cost, schedule, and quality. The volume of ECPs and the percent of undefined requirements are key indicators. It is often difficult to find a sufficient number of qualified personnel, and a shortage of such people can severely impact the schedule. Projects should closely monitor the number of open positions.

2. The requirements metrics will indicate the amount of engineering work still to be done to satisfactorily complete the PDR phase of the software development. Judgments as to the severity or criticality of the requirements that are not sufficiently defined to be testable or are not traceable are the essence of this indicator. The metrician, engineer, or manager should be skilled and experienced enough to take the quantitative measures of the metrics and translate them into a requirements risk assessment. Each application within a product division should have measures of when the number of requirements is sufficiently undefined or untraceable to raise the development to a medium to high risk level. It should be recognized that some projects or programs with 90 percent of the requirements testable and traceable may be very low risk while another program with a similar profile would be very high risk.

3. In general, if 15 to 20 percent of the requirements are not testable, then the development of the product or system should be considered medium to high risk. All requirements should be traceable at the conclusion of the PDR phase or a similar time slice.

4. Testability requires that every requirement be written such that test criteria can be established to yield yes or no (pass or fail) responses.

5. The units affected by ECPs are expected to increase for most ECPs. Many ECPs and a large number of affected units indicate a program that has not established firm requirements before initiating a contract.

6. The open (unresolved) review Action Items and requirements discrepancies are expected to spike upward at each review (such as SSR, PDR, and CDR) and then exhibit substantial decreases as workoff occurs after the review. Programs that issue clearly written specifications will experience
spikes that are low; programs that have good communications among the program office, systems engineering, and the customer will have a high rate of decay to this curve.

7. Requirements changes are expected early in a project as details of a system's operations are being defined and understood. At some point allocated requirements must be frozen. The later this happens, the greater the impact on cost and schedule.

8. Requirements changes occurring after CDR have significant schedule impact, even if the change is the deletion of a requirement. At the very least, documentation for requirements, design, and test will require rework.

9. The normal shape of the staff profile is to grow through the design phases, peak through the coding and testing phases, and then gradually taper off as integration tests are successfully completed, as an example.

10. Generally, the shape of the experienced staff profile should be high during the initial stage of the project, dip slightly during CSU development, and then grow somewhat during testing. The ratio of total to experienced personnel should typically be near 3:1 and should seldom exceed 6:1. Insufficient experienced personnel may lead to poor quality, which will cause schedule slips in later phases.

11. Understaffing results in schedule slippage and, if not corrected, in a continuing rate of slippage. The Cost and Schedule indicators should show this slippage.

12. Adding inexperienced staff to a late project will seldom improve the schedule and often causes further delays. Adding staff with the relevant experience may improve the schedule. However, it may be difficult to staff up with more experienced personnel in the required time and gains in the far future (more trained personnel) may outweigh near term schedule problems. In either case, cost will increase as reflected in the cost indicators.

13. A program that is experiencing a high personnel turnover rate cannot maintain needed continuity. Losses that impair the project knowledge and experience base should be treated as project risks with cost and schedule impacts.

14. A sudden increase in unplanned personnel losses may indicate the onset of a morale problem.
9 ROLES AND RESPONSIBILITIES: METRICS INFRASTRUCTURE

When implementing a metrics program, whether at the company level or at a product level, it is essential to keep the implementation simple. The measures must be simple, logical, familiar, absolutely necessary, and repeatable. Training, consensus, and buy-in are required.

This section provides general guidelines for establishing a metrics program in an organization with an emphasis on key roles of each phase of implementation. The material presented here is extracted, with permission, directly from work done by the Systems Engineering Technology Group at IBM Federal Systems Company, Owego, New York.\(^\text{30}\)

A successful metrics program requires dedication and persistence by all involved and may be a long process, but the long term benefits will be significant. Once a metrics program has been established, measurements can be used for defect prevention in real time, assessing process improvements and the historical data used for future project planning. The measurements are made both objectively, through the use of algorithms, and subjectively, with knowledge and experience of program personnel. In either case, an effective metrics program is evolved as the set of working metrics is continuously evaluated and revised.

Establishment and implementation of a metrics program includes the following steps:

1. Establishing the goals and objectives of the metrics program.
2. Building a framework needed to support the program.
3. Developing a data collection package.
4. Developing the algorithms and models into useful metrics.
5. Collecting the measurement data.
6. Creating and maintaining a historical database.
7. Assessing and using metrics to plan, monitor, and improve the process and products.
8. Refining and evolving useful metrics.

9.1 Establishing the Goals and Objectives

The motivation to develop a measurement program is derived from the recognition that attributes such as inefficiency and low productivity exist in some projects, and the realization that identifying what makes a project successful increases the chances of being able to eliminate negative attributes. One goal of the measurement program is to accurately identify both negative and positive attributes of a project. Another goal is to analyze the problem components, take steps to eliminate them in existing projects, and be aware of them in future efforts. A third goal is to analyze the components of successful projects and use this data to incorporate similar components into future projects. Tracking this data over time provides a basis for decision making in the future. A deciding factor in whether or not a measurement program will achieve its goals is the hierarchy of support within the organization and the extent of commitment to the program.

---


INCOSE-TP-1995-002-01 (originally INCOSE TM-01-001)
GOALS OF MEASUREMENT

- Thoroughly explore the goals prior to defining metrics.
- Understand who will use the metrics and why.

9.2 Establishing a Framework

A measurement program can only be as strong as the framework that supports the effort. Such a framework must begin with the initial approval and commitment from executive management. It is a struggle to start a measurement program, and it is not likely to be given enough priority unless it is also management’s priority. Measurement must become an integral part of the business and therefore must be part of, and flow down from, the corporate strategy.

To ensure continued progress within the measurement program, there needs to be a driving force behind the effort. Direction and motivation are ideally provided by an Engineering Process Group (EPG) whose responsibilities include the oversight of methods, tools, technologies, and measurements within the organization. The EPG will be the unified voice for the site Metrics Working Group (MWG), a group consisting of representatives from each project. The EPG has the task of maintaining the historical database, reporting to management, and controlling the integrity of the program. The EPG refines metrics, communicates metrics, and incorporates suggestions from project MWG representatives. The MWG representatives will collect data, use the reports for assessment, and suggest improvements for the program. Hopefully, a process owner will have been appointed prior to beginning the measurement program. The process owner will establish a measurement strategy, approve metrics, and review measurement reports from the projects. Establishing a support structure for the measurement program, with a certain degree of formalism, makes the measurement program a tangible entity. Once the framework is in place, the burden lies with the EPG to develop a data collection package.

ESTABLISHMENT OF A FRAMEWORK

- Executive management support and process ownership are essential.
- Buy-in by all involved participants must eventually be achieved.
- An Engineering Process Group (EPG) must plan, support, and control the integrity of the program.
- A Metrics Working Group (MWG) collects measurement data, distributes reports, evaluates results, and refines measurements.
- Formalizing the support framework will provide the strength needed to support the measurements effort.

9.3 Data Collection Package

The data collection package will serve as a guide and valuable reference for all participants in the measurement program. The package consists of the following:

- Metrics, ground rules, and rationale
- Names, roles, responsibilities, and schedules
- Blank data collection forms
- Sample completed data collection forms
The task of formalizing a set of metrics is a difficult one. The selection of metric data and derivation of measurements is beyond the scope of this article. Sommerville suggests that to determine if any particular metric is a useful predictor of product quality requires quality management to evaluate the metric in a systematic way. This involves convincing management that a relationship exists between the metric and the product quality of interest. In order to do this, a model must be formulated to allow this prediction to be made. Questions such as what will be measured, how it will be measured, and in what units it should be collected (labor-months, dollars, weeks, etc.), as well as the justification for these, should be a part of the formalization of a working set of metrics.

This set should be clearly and concisely defined, thereby removing as much ambiguity as possible. Thought must also be given to how the data will be normalized in order to allow for comparison across programs.

Example: Project X has delivered 2 documents with 2 defects reported. Project Y has 50 documents delivered with a total of 10 defects. If one looks at the raw number of defects, Project Y looks worse, but in reality this may not be the case. If the number of defects is normalized against the number of documents delivered, a more accurate picture can be drawn. This data can be normalized in other ways, such as against the number of pages or requirements. Careful consideration and experience gained through trial and error will help to determine how raw data should be normalized.

Specifying the names, roles, and responsibilities of those persons involved with measurement acknowledges the program’s credibility. It is a good idea to include schedules for reporting data as well. It should be noted however, that although data is formally reported at discrete points, the collection of data needs to be an ongoing, integral part of the process for the measurement program to be successful.

Another key component of the package is the data collection forms. They provide for the organized recording of pertinent data and the easy transmittal of the data into the historical database. Well designed forms will encourage completeness and correctness during the collection process. It is important that the forms be as self explanatory and stand-alone as possible.

Once the data collection package is complete, it is a good idea to test it on one or two projects. The purpose of this is to clarify definitions, refine measures, and change the forms as necessary. This will provide valuable input to the EPG and allow for improvements in the package. It is extremely important that attention be given to developing the package at the front end of this effort to ensure the understandability and readability of its contents. The amount of effort expended creating and refining the package will have a significant influence on the degree of difficulty of the actual data collection.

**DEVELOPMENT OF A DATA COLLECTION PACKAGE**

- A data collection package must be developed prior to implementing the program.
- Effort expended in developing it will reduce problems associated with collection.
- Well designed data collection forms encourage completeness and correctness.
- The package should be tested on one or two projects, and refined prior to introduction to all projects.

---

9.4 Collection of Data

Collection of data can be the most challenging aspect of a measurement program and therefore should be carefully planned and controlled. The EPG initiates the data collection and must control its growth rate. The first step in the collection process is to elicit management's support in assigning an MWG representative for each project from which data will be accumulated. This representative should be knowledgeable about all facets of the project.

Once the project MWG representative is selected, the EPG schedules a data collection kickoff meeting with this representative and the project leader. It is important to establish a specific day and time for the meeting to ensure it is not overlooked. Prior to this meeting, make sure the MWG representative is aware of the data that will be collected. Since data will be gathered at this meeting, it is necessary that the data collection package be completed and ready to use.

Although the package is functional at this point, collection of measurement data should be introduced in phases. Many changes will be occurring at the front end of the measurement program so it should be relatively stable before taking on too many projects. Begin with a limited set of data to be collected from a few small projects. When this is complete, add more detail and depth to the set of metrics and collect from more projects, requesting support as needed. When the effort is in full swing, data can be collected from all active projects.

The initial collection will be up to the EPG, with responsibility transitioning to the appointed MWG representative on each project. Data will be collected continuously so it is important to clearly define the points at which it will be formally reported. The raw data collected is valuable, but it must be incorporated into a tracking utility for it to be useful. This tracking utility will become the historical database that will allow for computation, normalization, and analysis.

COLLECTION OF DATA

- Data collection must be carefully planned and controlled.
- Collection should be phased in, beginning with a couple of projects, and then expanded to include all projects.
- The EPG must kick off the collection and transition the responsibility to the project MWG representatives.
- Data must be collected continuously but reported at discrete control points.
- Raw data must be transformed through computation and/or normalization to usable metrics.

9.5 Creation of a Historical Database

Measurement data will be stored in a historical database, which provides the capability for data to be accumulated long term in an organized manner. An individual, preferably in the EPG, experienced with databases should be tasked with creating a prototype. The database fields must be carefully defined to ensure the types, sizes, and units correspond to the data being collected. Since this is an initial effort there are many software packages available that will be sufficient to meet the needs of the prototype database. Its basic capabilities must include easy accommodation of future changes and generation of reports in textual and graphical format. Use of the prototype will provide the knowledge base needed to plan for the automated metrics system that will be required as the effort reaches toward maturity.
Once the database is created and the data has been collected, the data can be input into the database. Upon completion, usable reports can be generated and distributed. These reports can then be analyzed, providing the potential to detect trends, both negative and positive, in the process and its products. Since large quantities of important data will be stored in this utility, a database administrator should be appointed to protect its integrity.

**CREATION OF A HISTORICAL DATABASE**

- Carefully design the database to allow change and growth.
- Choose a sufficient software package for prototyping.
- Design the format of reports in parallel with the structure of the database.
- Capture lessons learned with the prototype database for future use in planning for a fully automated system.

9.6 Using, Refining, and Evolving the Metrics

Although the benefits of having an active measurement program are many, several challenges will be encountered along the way. A common problem, encountered by all startup efforts that require a modification in the process, is the individual’s resistance to change. This may be due in part to the fact that the added value of exerting effort to collect data has not yet been sufficiently demonstrated, thus manifesting critics and skeptics. There may also exist the concern that people, instead of the process and products, are being measured. Also, the initial effort may involve collection of some data that is difficult, if not impossible, to obtain.

The intangibility of the data is one of numerous uncertainty factors that will increase the complexity of the collection. Presently it is uncertain what constitutes a good, useful metric. There is some speculation, however that there needs to be further experimentation before anything can be proven. The way in which the data is collected is also subject to debate. Ideally, all measurement data would be collected objectively. This would ensure that given an algorithm, two individuals obtaining metrics on a particular process or product aspect would yield the same result time and again, thus confirming that the metric is “correct.” Some metrics, however, do not have algorithms that can be applied to them. This necessitates the use of subjective measuring to obtain data. This technique, which relies heavily on knowledgeable individuals making judgments based on personal experience, is characteristically noted for discrepancies.

Yet another challenge lies in the fact that there are variations in the system development process across projects since each project typically emphasizes different subprocesses in its execution due to the differences in product mix. This aspect makes analysis across programs very difficult.

Despite those and countless other challenges that lie ahead, the future is bright for the measurement discipline. The working set of metrics will continue to be refined as we gain experience and acquire more knowledge. There will be a push to automate the collection of data, therefore making the integration of measurement collection into the process much easier and less time consuming.

**FACING THE CHALLENGE**

- Significant challenges exist for measurement of the system development process and its products.
- It is important that the challenges faced be recognized and incorporated into the overall program and its goals.
Key factors that contribute to the success of a measurement program include careful planning, control, and a framework for support. All challenges and potential impacts must be explored and minimized to make measurement as unobtrusive as possible. A high degree of control is needed to ensure the program is followed and data integrity is maintained. As we actively work to build and mature our measurement programs, sharing information within and between corporations will be an invaluable way to make advances in the measurement field.

9.7 Proven Ideas for Successful Application

- Only with effective measurement can companies improve the way they do business (the process) and the quality of their products.
- Goals and Objectives must also be identified for smaller groups, teams, or projects. When goals are defined, the right questions can be asked to determine what measurements should be implemented for a particular situation.
- When first implementing a metrics program, start with a well defined, small set of metrics. Once established, only then evolve to a more sophisticated set of metrics through refinement, adding new metrics to the list, and taking away those that no longer apply.
- Buy-in from all levels (team, group, project, etc.) is a “must have” – always! As a part of the buy-in process, the team or group must be able to participate and make some choices in the selected metrics. We have identified core sets of metrics for each engineering discipline through a series of workshops and discussions. When we use these core metrics on a project, the entire project team should help tailor the core set (add, modify or take away from) for that project’s needs – part of the buy-in process.
- Training (critical). Awareness training should be first to educate team members to the importance of measurement, how to implement (collect, analyze, and report) the selected metrics, and most importantly, what to do with the analyzed and reported data (how to determine what action is required (if any) based on the results).
- Don’t measure the people (essential), measure the process and the quality of the product. From various companies’ experiences, personnel do not like to be measured and tend to become paranoid even when they believe in measurement. Measurement can easily become a negative force, but it does not need to be. Knowledge through training plus buy-in help reduce the negatives.
- Know your audience when presenting metrics (for metrics enthusiasts everywhere) – give the details to the analyst – give the bottom line to the manager (or driver). Tell management what the metrics report means as related to the project, schedule, and cost. Tell the analyst how the results were derived.
- “Keep it simple, stupid.” This repeated little saying applies to many areas related to measurement. One in particular, not previously discussed, is documentation. Don’t make it so complicated that readers need a doctorate in mathematics to understand it. Write it for the user, not the analyst.

9.8 Management Support and Commitment

For successful implementation of any metrics program, there are five important aspects to consider:

1. Managers must be involved and committed and should take the lead in initiating any measurement program. Management is responsible for defining goals and objectives and asking for feedback information.
2. Sufficient budget for measurement tasks must be clearly allocated so that the responsible individual(s) will be able to put in the required effort for completion.
3. Each project or organization must have a metrics “champion” to promote the cause, to write the implementation plan, to organize the effort, and to distribute the information according to the needs of the participants.

4. The work environment must be goal oriented – there is no point in promoting or implementing measurement without a strategy.

5. Training is essential to overcome old mind sets, emotive issues, and fear of being scrutinized or criticized. Collaboration is crucial. Regardless of how structured and logical your strategy, without the commitment of your staff members, all plans are open for failure. Therefore, training is particularly essential here during all levels of the implementation.

9.9 Metrics Council or Center of Excellence

It is recommended that any company interested in implementing a metrics program also establish its own internal Engineering Metrics Working Group. An internal Metrics Working Group should be multifunctional, consisting of representatives from each of the disciplines, projects, and programs. A key effort is to capture and promulgate metrics lessons learned throughout government and commercial environments.

9.10 Metrics Database

As a part of implementing a metrics program, define early on how the collected metrics data will be stored. For example, will a distributed database be used? Will the data be kept on a shelf? Or will the data be maintained online in an Excel format or other application? Within the constraints of classified and proprietary information, most performing organizations have a long term goal to provide a central database dedicated for storing historical and current metrics data.
**ACRONYMS**

<table>
<thead>
<tr>
<th>ACRONYM</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Analysis</td>
</tr>
<tr>
<td>ACWP</td>
<td>Actual Cost Work Performed</td>
</tr>
<tr>
<td>AOC</td>
<td>Award of Contract</td>
</tr>
<tr>
<td>BAC</td>
<td>Budget at Completion</td>
</tr>
<tr>
<td>BAFO</td>
<td>Best and Final Offer</td>
</tr>
<tr>
<td>BCWP</td>
<td>Budgeted Cost of Work Performed</td>
</tr>
<tr>
<td>BCWS</td>
<td>Budgeted Cost of Work Scheduled</td>
</tr>
<tr>
<td>BOE</td>
<td>Basis of Estimate</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CAE</td>
<td>Computer Aided Engineering</td>
</tr>
<tr>
<td>CC</td>
<td>Circuit Card</td>
</tr>
<tr>
<td>CCA</td>
<td>Circuit Card Assembly</td>
</tr>
<tr>
<td>CDR</td>
<td>Critical Design Review</td>
</tr>
<tr>
<td>CDRL</td>
<td>Contract Data Requirements List</td>
</tr>
<tr>
<td>CE</td>
<td>Concurrent Engineering</td>
</tr>
<tr>
<td>CEL</td>
<td>Capital Equipment List</td>
</tr>
<tr>
<td>CER</td>
<td>Capital Expenditure Request</td>
</tr>
<tr>
<td>CMI</td>
<td>Capability Maturity Index</td>
</tr>
<tr>
<td>CMM</td>
<td>Capability Maturity Model</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial-Off-The-Shelf</td>
</tr>
<tr>
<td>CSC</td>
<td>Computer Software Component</td>
</tr>
<tr>
<td>CSE</td>
<td>Chief System Engineer</td>
</tr>
<tr>
<td>CSCI</td>
<td>Computer Software Configuration Item</td>
</tr>
<tr>
<td>CSU</td>
<td>Computer Software Unit</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-The-Shelf</td>
</tr>
<tr>
<td>D</td>
<td>Demonstration</td>
</tr>
<tr>
<td>EAC</td>
<td>Estimate at Completion</td>
</tr>
<tr>
<td>ECO</td>
<td>Engineering Change Order</td>
</tr>
<tr>
<td>ECP</td>
<td>Engineering Change Proposal</td>
</tr>
<tr>
<td>EMC</td>
<td>Electromagnetic Control</td>
</tr>
<tr>
<td>EMI</td>
<td>Electromagnetic Interference</td>
</tr>
<tr>
<td>EPG</td>
<td>Engineering Process Group</td>
</tr>
<tr>
<td>ESC</td>
<td>Estimated Schedule to Complete</td>
</tr>
<tr>
<td>ESLLOC</td>
<td>Equivalent Source Lines of Code</td>
</tr>
<tr>
<td>ETC</td>
<td>Estimate To Complete</td>
</tr>
<tr>
<td>FCA</td>
<td>Functional Configuration Audit</td>
</tr>
<tr>
<td>GQM</td>
<td>Goal/Question Metric</td>
</tr>
<tr>
<td>HF</td>
<td>Human Factors</td>
</tr>
<tr>
<td>H</td>
<td>Hardware</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interface</td>
</tr>
<tr>
<td>HP/CR</td>
<td>Hardware Problem/Change Report</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>HWCI</td>
<td>Hardware Configuration Item</td>
</tr>
<tr>
<td>I</td>
<td>Inspection</td>
</tr>
<tr>
<td>ACRONYM</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>INCOSE</td>
<td>International Council on Systems Engineering</td>
</tr>
<tr>
<td>IPD</td>
<td>Integrated Product Development</td>
</tr>
<tr>
<td>IPT</td>
<td>Integrated Product Team</td>
</tr>
<tr>
<td>IRS</td>
<td>Interface Requirements Specification</td>
</tr>
<tr>
<td>KLOC</td>
<td>Thousand (K) Lines of Code</td>
</tr>
<tr>
<td>LH</td>
<td>Labor Hours</td>
</tr>
<tr>
<td>LCL</td>
<td>Lower Control Limit</td>
</tr>
<tr>
<td>LOC</td>
<td>Lines of Code</td>
</tr>
<tr>
<td>LOE</td>
<td>Level of Effort</td>
</tr>
<tr>
<td>LSL</td>
<td>Lower Specification Limits</td>
</tr>
<tr>
<td>MWG</td>
<td>Metrics Working Group</td>
</tr>
<tr>
<td>OPC</td>
<td>Overall Proportion of Code</td>
</tr>
<tr>
<td>PAT</td>
<td>Process Action Team</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>PCI</td>
<td>Product Complete Indicator</td>
</tr>
<tr>
<td>PEM</td>
<td>Product Engineering Manager</td>
</tr>
<tr>
<td>PDR</td>
<td>Preliminary Design Review</td>
</tr>
<tr>
<td>PMO</td>
<td>Program Management Office</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>QFD</td>
<td>Quality Function Deployment</td>
</tr>
<tr>
<td>RAM</td>
<td>Reliability and Maintainability</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>ROM</td>
<td>Rough Order of Magnitude</td>
</tr>
<tr>
<td>RMR</td>
<td>Request for Material Review</td>
</tr>
<tr>
<td>S</td>
<td>Software</td>
</tr>
<tr>
<td>SCN</td>
<td>Specification Change Notice</td>
</tr>
<tr>
<td>SDR</td>
<td>System Design Review</td>
</tr>
<tr>
<td>SE</td>
<td>Systems Engineering</td>
</tr>
<tr>
<td>SLOC</td>
<td>Source Lines of Code</td>
</tr>
<tr>
<td>SP/CR</td>
<td>Software Problem/Change Report</td>
</tr>
<tr>
<td>SPR</td>
<td>Software Problem Report</td>
</tr>
<tr>
<td>SQA</td>
<td>Software Quality Assurance</td>
</tr>
<tr>
<td>SRR</td>
<td>System Requirements Review</td>
</tr>
<tr>
<td>SRS</td>
<td>Software Requirements Specification</td>
</tr>
<tr>
<td>SS</td>
<td>System Specification</td>
</tr>
<tr>
<td>SSR</td>
<td>Software Specification Review</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>T</td>
<td>Test</td>
</tr>
<tr>
<td>TP/CR</td>
<td>Test Problem/Change Report</td>
</tr>
<tr>
<td>TPM</td>
<td>Technical Performance Measure</td>
</tr>
<tr>
<td>TQM</td>
<td>Total Quality Management</td>
</tr>
<tr>
<td>TRR</td>
<td>Test Readiness Review</td>
</tr>
<tr>
<td>UCL</td>
<td>Upper Control Limit</td>
</tr>
<tr>
<td>USL</td>
<td>Upper Specification Limit</td>
</tr>
<tr>
<td>WAD</td>
<td>Work Authorization Document</td>
</tr>
<tr>
<td>WBS</td>
<td>Work Breakdown Structure</td>
</tr>
</tbody>
</table>

INCOSE-TP-1995-002-01 (originally INCOSE TM-01-001)
GLOSSARY OF TERMS

Category
A general class or functionality such as hardware, software, test, or systems. For metrics, category implies a grouping of like measures. Varies dependent upon discipline and metric/measure (e.g., hardware categories: digital, analog, RF, etc.)

Collection Points
Periodic collection times for those items to be evaluated at standard control points. The collection points extend throughout the system’s life cycle.

Defect
Any flaw found in a specification, design, or implementation of a product.

Defect Severity
Major: A major error will cause a malfunction or is a deviation from the requirements or intended specifications.
 Minor: A minor error indicates a violation of conventions or rules that could result in difficulties in terms of maintenance or clarity of purpose.
 Trivial: A trivial error is one that is a typographical or spelling error whose correct interpretation is obvious to a knowledgeable reader and has no effect on the product’s functions.

Defect Types
Wrong: A wrong error is one that is attributed to a technical or content error, a violation of the standards related to the inspected product or documentation, or any other error that does not fit the missing or extra categories.
 Missing: A missing error is one that is attributed to missing information, i.e., requirements, assumptions, limitations, and constraints.
 Extra: An extra error is one that is attributed to repeated or irrelevant information.

Inspections
An internal, formalized, technical review performed by a small group of people (usually a knowledgeable set of peers). The main purpose is to find errors in intermediate work products and documentation as early in the development process as possible.

Measurement
Process of collecting quantitative data regarding processes, performance, or product quality.

Metric
A composite of meaningful measures taken over time that communicate important information about processes, performance, or quality.

Producers
Developers who create work products.

TPMs
Key program technical performance measures (system performance, capacities, user response times, etc.) being tracked and reported.

Type
A lower taxonomic category or sub-tier to category such as reused, new, modified as applied to software code. Another example is requirements types such as performance, timing, security, etc.

Users
Developers who use work products.
Work products: Products or by-products of the development process to include code, design, and documents.
BIBLIOGRAPHY


Bibliography (Continued)


Bibliography (Continued)


Bibliography (Continued)

[Lim, Wayne C.] Reuse Metrics Overview and Recommendations: Work in Progress, Version 0.0, Hewlett-Packard Corporate Engineering.


Bibliography (Continued)


**Bibliography (Continued)**


APPENDIX A -
GUIDEBOOK FUTURES

Volume II of the Guidebook will contain sections representing INCOSE cross-functional working group projects in the following areas:

1. Metrics Catalog
2. Executive Metrics
3. Risk-Based Metrics
4. A Metrics Starter Kit
5. Organizational Metrics
6. System Engineering Metrics Derived from Benchmarking
7. Metrics and the Systems Engineering Capability Maturity Models
8. Metrics and Subcontract/Supplier Performance
9. Metrics Tools