Are We Doing Well, Or Are We Doing Poorly?

Linda Westfall
The Westfall Team

Abstract

Software metrics don't solve problems – people solve problems. What software metrics can do is provide information so you can make informed decisions and better choices. According to the new ISO/IEC 15939 Software Engineering -- Software Measurement Process standard, decision criteria are the "thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result". In other words, you need decision criteria to obtain guidance that will help you interpret the measurement results. This paper shows you how to establish useful decision criteria for different types of metrics.

Introduction

Measurement is common in our daily lives. We use measurement all of the time to help us make better decisions based on facts and information. For example, one of the first things you probably did today was measure. The alarm clock went off and you looked at the time. You Measured! Then you made a decision based on that measurement, "Do I have to get up right now or can I hit the snooze alarm and sleep for ten more minutes?" That's the objective of software measurement as well, to provide software managers and software engineers with the information they need to make better decisions based on facts rather than on opinion or "gut feel."

One of my favorite metric cartoons shows a metrics analyst presenting a variety of charts and graphs at a meeting. The caption has the boss asking, "Norman, please – just tell us, are we doing good, or are we doing bad?" In other words, unlike the simple decision of getting out of bed, in the complex arena of software engineering, it's not always so easy to interpret the software measurements and use them in decision-making. This is where defining the decision criteria for our metrics helps. We establish decision criteria to help us interpret the results of the measurement then we use them in making our decisions.

According to Watts Humphrey, there are four major roles for software measurement: [Humphrey-89]

- *Control:* Control type metrics are used to monitor our software processes, products and services and identify areas where corrective or management action is required.

- *Evaluate:* Evaluate type metrics are used in the decision-making process to study products, processes or services in order to establish baselines and to determine if established standards, goals and acceptance criteria are being met.

- *Understand:* As part of research studies, understand type metrics can be gathered to learn about our software processes, products and services.

- *Predict:* Predict type metrics are used to estimate the values of base or derived measures in the future.

We will use these different types of roles to explore establishing decision criteria for different types of measures.

Measurement Information Model

So where do decision criteria fit into the measurement process? To answer that, let's step back and look at the Measurement Information Model as defined in ISO/IEC and illustrated Figure 1. This model links the relevant entities and their attributes to an information product that matches the information needs of the measurement's user (customer).
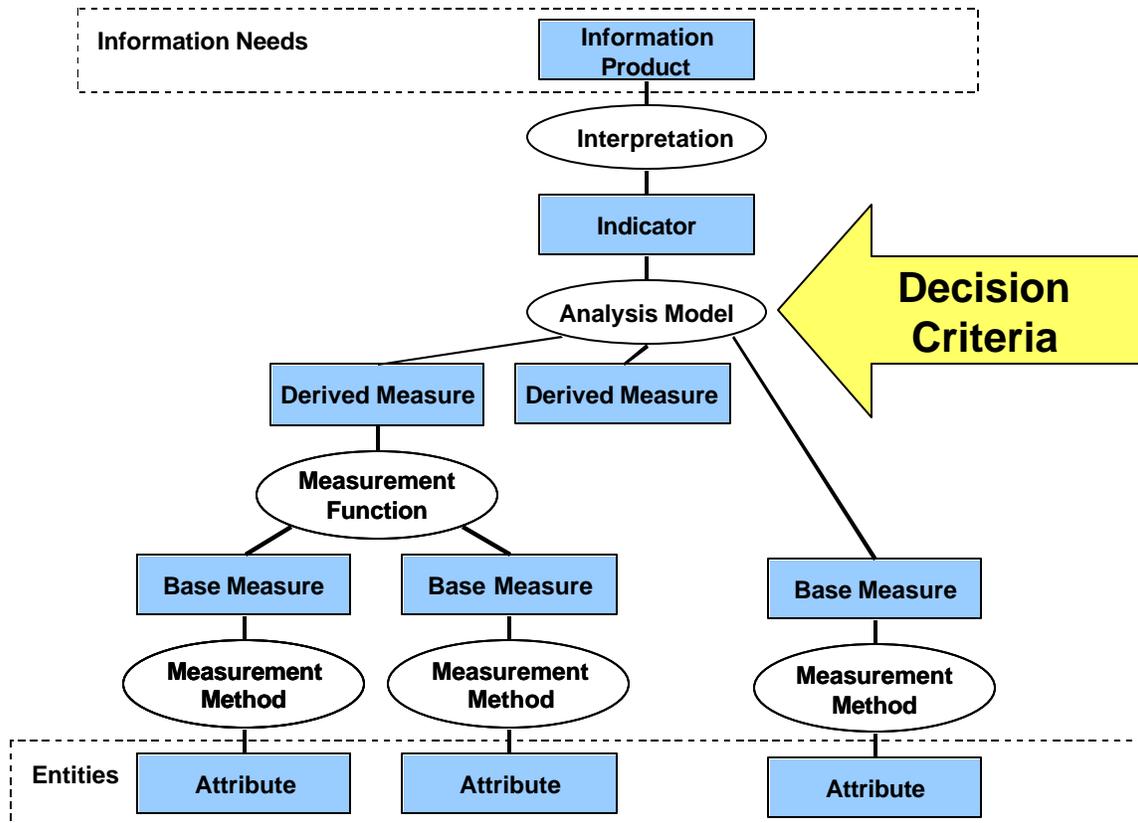

Figure 1 – Measurement Information Model

Entities & Attributes: An entity is a specific object (e.g., software work product, software process or resource) that has specific measurable attributes (properties or characteristics). For example, if the software source code was the entity we might want to measure the attributes of its size or the number of defects found in that code.

Measurement Methods & Base Measures: The measurement method includes the defined rules for mapping numbers or symbols onto the measured value to create a base measure for an attribute. For example, we might count semicolons to assign a base measure in logical lines of code to the size attribute for module X.

Measurement Functions & Derived Measures: A measurement function is the algorithm or calculation used to combine two or more base measures into a derived measure. For example, the measurement function of dividing the number of defects in module X by the number of lines of code in module X and multiplying by 1000 (to convert to defects per thousand lines of code) could be used to calculate the defect density for module X.

Analysis Model: According to ISO/IEC 15939, an analysis model is "an algorithm or calculation combining one or more base and/or derived measures with associated decision criteria. It is based on an understanding of, or assumptions about, the expected relationship between the component measures and/or their behaviors over time." For example, based on historic data we could calculate the mean and standard deviation defect density for a set of source code modules

in a similar product that met its post release quality and reliability requirements and use those to model an acceptable range of defect densities for the current set of modules under consideration.

Decision Criteria: According to ISO/IEC 15939, "decision criteria may be calculated or based on a conceptual understanding of expected behavior. Decision criteria may be derived from historical data, plans, and heuristics, or computed as statistical control limits or statistical confidence limits." For example, based on our historic model of "acceptable" defect density, we might establish decision criteria that states that any module with a defect density greater than one standard deviation above the mean is a candidate for reinspection before release to integration test. Any module with a defect density greater than two standard deviations above the mean would require further investigation as a candidate for possible reengineering.

Indicators, Interpretation and Information Products: The application of the analysis model and decision criteria results in an indicator that "provides an estimate or evaluation of specified attributes derived from an analysis model with respect to the defined information needs." One or more indicators, along with their interpretations, constitute an information product that addresses the information need. Figure 2 illustrates the indicator that might be used for this defect density example.

Decision Criteria for Control Type Metrics

Control type metrics are used to monitor our software processes, products and services and identify areas where corrective or management action is required. Control type metrics act as "red-flags" to warn us when things are not as expected or planned. If all is going well, the decision should be "everything is fine and therefore no action is required." The decision criteria for control type metrics usually take the form of:

- Thresholds
- Variances
- Control limits

Thresholds: Thresholds are established boundaries that when crossed indicate that action or investigation is needed.

Thresholds may be established based on historic data like in the defect density metric discussed in the above example. In this example, two thresholds were established based on calculating the mean and standard deviation from historic defect density

Figure 2: Example – Defect Density Indicator

values. Similar graphs could be established for thresholds based on industry standards or benchmarks. For example, empirical evidence shows that modules with a McCabe's Cyclomatic Complexity greater than 10 are more error prone and harder to maintain. [Jones-91] We could create a graph similar to the one in Figure 2 where the thresholds are based on the acceptable values being a Cyclomatic Complexity <=10, the cautionary values being from say 10 to 20 and the unacceptable values being > 20.

Threshold may be established based on future predictions. For example, if while planning our project we predict a 15% staff turnover rate, we will define our project's staffing and training plans accordingly. However, a risk exists that there may be a higher turnover rate than 15% that will impact our ability to complete the project on schedule and/or deliver the promised level of functionality or quality.
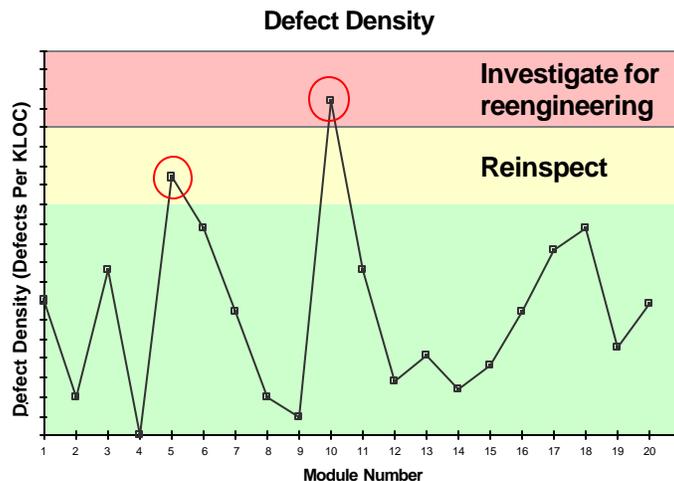
We can utilize this staff turnover measure as a risk management trigger. The decision criterion is therefore very straight forward, if the 15% threshold is exceeded; implement the staffing and training risk contingency plan. The run chart illustrated in Figure 3 could be used to track this measurement over time against its specified threshold.

Similar graphs could be utilized to track other risk triggers based on predicted values used to plan the project. For example, we could track the actual requirements churn (scope creep) against predicted values, actual productivity level against the levels used to predict staffing requirements or actual product size against the size estimates used to predict project effort and schedules.
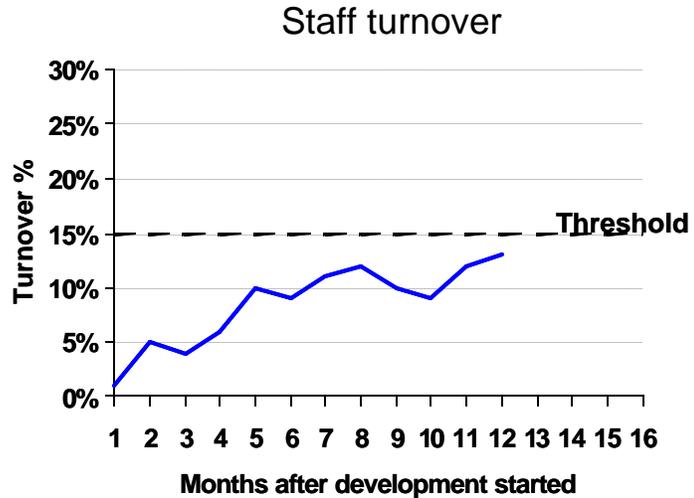
Staff turnover

Figure 3: Staff Turnover

Thresholds may be established based on customer requirements. For example, there may be customer requirements for capacity, throughput, response times, memory utilization, availability, reliability or quality levels. These types of technical system requirements must be designed and built into the product. If the actual measures during modeling or testing do not meet the requirements (i.e., the threshold is passed) then a defect report should be opened to investigate the need for a product design change.

Another example of customer requirements being used as a threshold is illustrated in Figures 4 and 5. Here the customer had a service level agreement requirement that 95% of all major defects reported from the field would be corrected with 30 days. The run chart in Figure 4 tracks the percentage of problem reports closed each month within the 30-day limit. The problem with this graph was that it was a lagging indicator. The decision criteria for this measure was to apply more resources to problem resolution during the next month, which might help the subsequent month's measurement value if they were applied correctly, however this does not correct the fact that we missed the requirement during the current period. We augmented this metric with the second distribution graph in Figure 5. This graph and its associated 30-day threshold allowed us to be more proactive in meeting our service level agreement requirements. The decision criteria
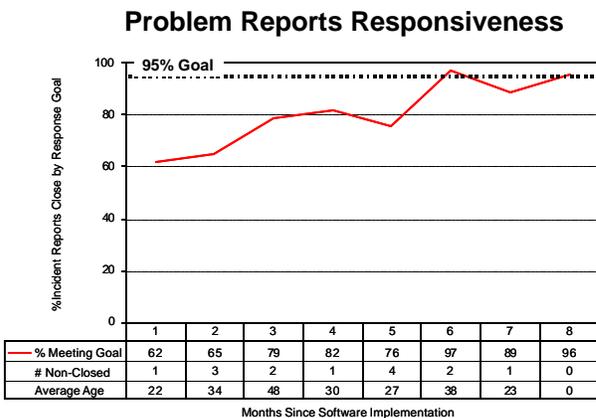
Problem Reports Responsiveness

	1	2	3	4	5	6	7	8
% Meeting Goal	62	65	79	82	76	97	89	96
# Non-Closed	1	3	2	1	4	2	1	0
Average Age	22	34	48	30	27	38	23	0

Figure 4: Problem Report Responsiveness
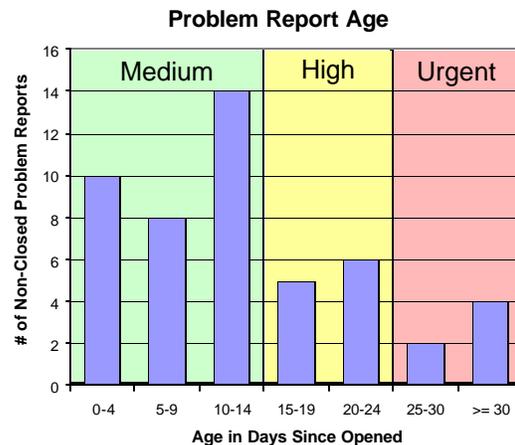
Problem Report Age

Figure 5: Problem Report Age

for this age measure was to prioritize our defect resolution activities on defects that were approaching 30 days old.

Variances: Variances compare actual values with expected values and make decisions based on the magnitude of the difference. Many times the expected value is based on estimates or predictions. Variances are typically calculated in one of two ways, either as a ratio (actual value divided by expected value) or as an absolute delta (actual value minus expected value).

The decision criteria ranges for ratio type variances are typically established at 1 (or 100%) plus and minus a value representing an acceptable level of variation. Note that a variance of 1 (100%) indicates that the actual matches the expected value at that specific point in time.

Resource Utilization

Figure 7: Resource Utilization

The Assess Status of Coding Activities example in Annex A of the ISO/IEC 15939 standard uses a ratio type variance by dividing the Coding Units Planned to Date multiplied by 100 into the Progress to Date; where the Progress to Date is the sum of the percentage complete for each module. Figure 6 shows an example graph of what this ratio type variance measure might look like.
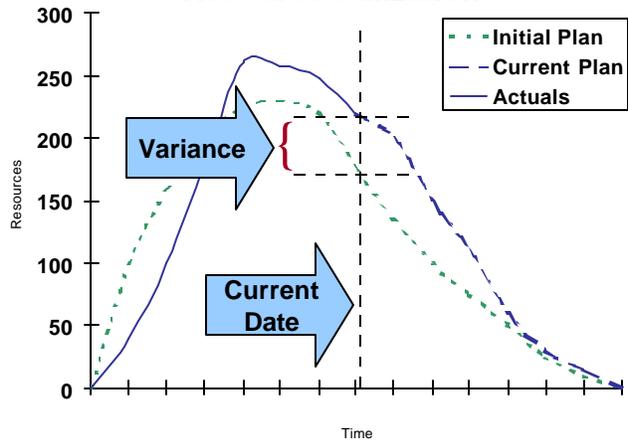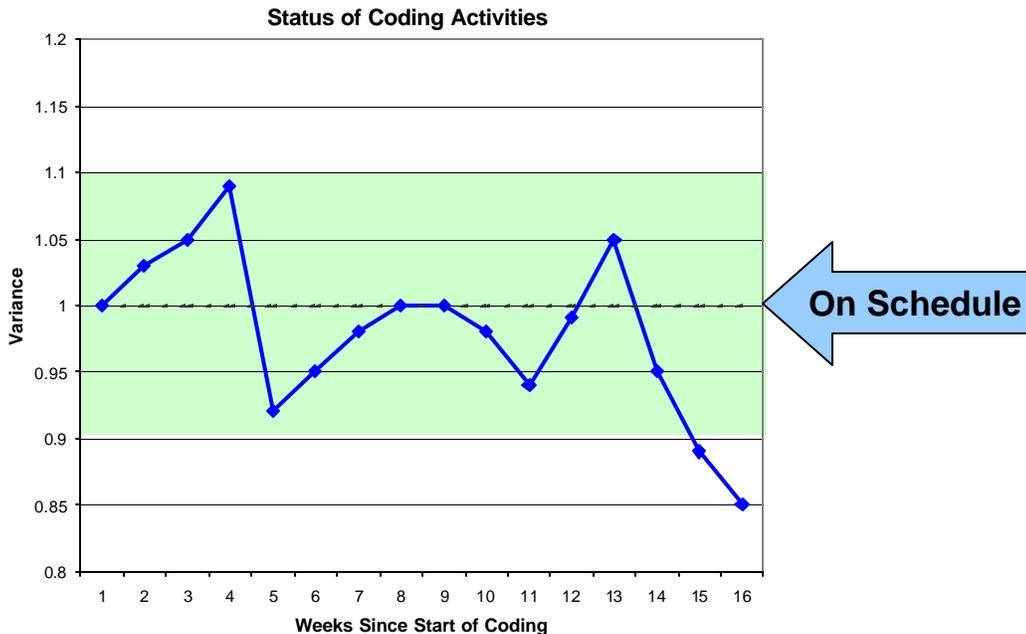
Figure 6: Assess Status of Coding Activities

A ratio type variance is influenced by the changes in the magnitude of the actual and expected values over time. For example, if early in the project we compare the number of tasks actually completed to date vs. those estimated to be completed, each individual task carries a large impact on the variance. If we expect 10 tasks to be complete and one task is late, the variance is 9/10 or 90%. If we look at the variance later in the project, for example when we expect 100 tasks to be complete and one task is late, the variance is 99/100 or 99%. Depending on the

magnitude of the estimate and actual at any given point in time, the same absolute delta between those two values will result in different ratio variances.

Early in a process or project it may also be more acceptable (i.e., less risky) to have larger variances because there is plenty of time left to make adjustments and bring the variance under control. But as the process or project nears its completion date the acceptable amount of variance may be much smaller.

For these two reasons, management may want to consider setting different variance ranges for different times in the process or project. For example, during a one year long project the initial acceptable variance might be set to 100% ± 25% and the cautionary range set to 100% ± 40%. By mid-project, the acceptable variance might be reduced to 100% ± 15% and the cautionary range reduced to 100% ± 25%. Finally, late in the project, acceptable variance might again be reduced to 100% ± 5% and the cautionary range reduced to 100% ± 10%.

One of the advantages of using an absolute delta type variance is that it avoids the issue of the influence of the magnitude of the actual and expected values over time. Figure 7 shows an example of a resource utilization metric using absolute delta variances. Another advantage of using absolute delta variances is that for cumulative measures like tasks completed, effort expended, resources utilized or costs, the expected values typically planned out to the end of the process or project and the actuals to date can be extrapolated

Establishing decision criteria for a variance is typically a subjective judgment based on what management determines to be "in control". Considerations when establishing these decision criteria include:

- *Historic Values:* Collecting and analyzing historic information on past variances for similar processes, products or services can provide valuable information about the capabilities of the current estimation/prediction processes or what acceptable variances were historically when we had successful processes, products or services. Current capabilities and past successes should be taken into consideration when establishing decision criteria.

- *Customer, Contractual or Regulatory Requirements:* There may be specific customer, contractual or regulatory requirements that dictate acceptable/cautionary/unacceptable values for certain variances.

- *Best Practices and Workmanship Standards:* Our organizations may also have established best practices or workmanship standards that dictate acceptable/cautionary/ unacceptable values for certain variances.

If the measurement value for a variance is within the acceptable level, no action is required. If it is in the cautionary level it indicates that at a minimum the measure should be monitored more closely to ensure that it returns to an acceptable value. We may also want to do an analysis of the individual base or derived measures we are using to determine the cause(s) of their less than optimum values and take corrective action as necessary. If the measurement value for a variance is in the unacceptable range, an analysis must be conducted to determine the cause(s) and corrective action should be taken.

Control Limits: Statistical process control charts with control limits are one of the classic ways of controlling processes. To establish control limits, the mean and standard deviation are calculated from a set of sample data from the process. The mean becomes the center line (CL) and zones are created at ±1, ±2 and ±3 standard deviations from the mean. The upper control limit (UCL) is set at plus 3 standard deviations above the mean. The lower control limit (LCL) is set at minus 3 standard deviations below the mean (or at zero if this value is below zero and negative numbers are not valid measures).

As illustrated in Figure 8, the decision criterion is based on 5 patterns:

- Pattern 1: A single data point more than ±3 standard deviations from the mean

- Pattern 2: Two out of three data points in a row more than ±2 standard deviations from the mean

- Pattern 3: Eight successive data points on the same side of the centerline

- Pattern 4: 4 out of five data points in a row more than ±1 standard deviations from the mean

- Pattern 5: A run of 8 successive data point in a row either all trending up or all trending down.

While these data patterns are all possible they are highly improbable. If one of these patterns is identified, the process that created the data should be investigated to determine if there is an assignable cause that needs to be corrected or if the data pattern is the result of normal variation in the process.
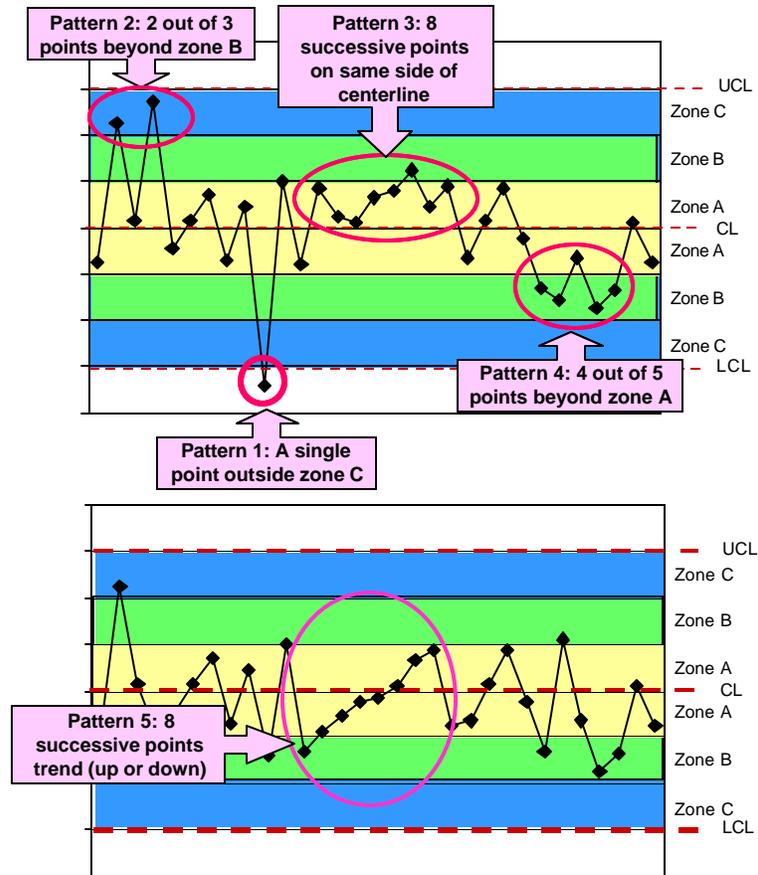

Figure 8: Control Charts Decision Criteria Patterns

Decision Criteria for Evaluate Type Metrics

Evaluate type metrics are used to examine and analyze our software processes, products and services as part of the decision-making process. Examples of using evaluate type metrics include:

- Performing cost/benefit analysis

- Analyzing and prioritizing choices

- Performing analysis to determine if a process is ready to start or end (entry & exit criteria)

Cost/Benefit Analysis: Cost/benefit analysis looks at the predicted the costs of a project or action and compares those against the predicted benefits of undertaking that project or action in order to:

- Determine if predicted return on investment (ROI) is sufficient enough to recommend initiating that project or action

- Select between alternative projects or actions

Typically the cost/benefit analysis is performed using a ratio calculated by dividing the benefit of a project or action by its cost. Examples of cost include the cost of people (salaries, benefits, travel, training, contractor's fees), materials, methods and tools, capital equipment, infrastructure,

and risks. Examples of benefits include future revenue, maintaining or increasing market share, improving technical leadership, increasing capability or capacity, and reducing risk.

Any benefit-to-cost ratio less than 1 indicates that the project or activity costs more than its expected benefit and should be rejected. However, we also need to make a profit on our investment, a benefit to cost ratio of 1 indicates only a break even. When establishing the decision criteria for a benefit-to-cost ratio we need to consider an acceptable profit margin. For example, if we determine that we need to make at least a 15% profit on our investment or the money should be used elsewhere then we would establish our decision criteria for the benefit-to-cost ratio at >= 1.15.

As a further example, consider a project risk that we have analyzed and determined to have a 25% likelihood of turning into a problem (the probability of an undesirable outcome) and if it turns into a problem we are predicting the loss will be $300,000. The expected value of the risk, its Risk Exposure (RE), is therefore 25% of $300,000 = $75,000. If our decision criterion for Risk Exposure indicates that if our exposure is over $50,000 for any given risk, we will explore risk reduction alternatives to reduce that risk. As illustrated in Table 1, we have come up with three alternatives to consider.

Risk #	Prob(UO) before	Loss (UO) before	RE before
143	25%	$300K	$75K

Alternative	Prob(UO) after	Loss(UO) after	RE after	Cost	RRL
1	5%	$300K	$15K	$65K	0.9
2	25%	$160K	$40K	$25K	1.4
3	20%	$300K	$60K	$2K	7.5

Table 1: Risk Reduction Leverage

Alternative #1 is expected to reduce the risk's probability of turning into a problem to 5%. If we implement this alternative, our new predicted risk exposure is now 5% of $300,000 = $15,000. This is a risk reduction benefit of $75,000 - $15,000 = $60,000. But this risk reduction activity is predicted to cost $65 so the benefit to cost ratio is $60,000 / $65,000 = .9. Since this is less than 1, we reject this alternative.

Alternative #2 is expected to reduce the risk loss if it turns into a problem to $160,000. If we implement this alternative, our new predicted risk exposure is now 25% of $160,000 = $40,000. This is a risk reduction benefit of $75,000 - $40,000 = $35,000. This risk reduction activity is predicted to cost $25,000 so the benefit to cost ratio is $35,000 / $25,000 = 1.4. Since this is less than our decision criteria of 1.15, this choice is still in the running.

Alternative #3 is expected to reduce the risk's probability of turning into a problem to %20. If we implement this alternative, our new predicted risk exposure is now 20% of $300,000 = $60,000. This is a risk reduction benefit of $75,000 - $60,000 = $15,000. This risk reduction activity is predicted to cost $2,000 so the benefit to cost ratio is $15,000 / $2,000 = 7.5. Clearly this alternative has the highest benefit to cost ratio of the three alternatives. But wait – remember our risk exposure decision criteria that said any risk exposure over $50,000 requires us to explore risk reduction alternatives to reduce that risk. The new risk exposure for this risk after the alternative #3 risk reduction action is taken is still at $60,000. Therefore, unless we implement both alternatives #2 and #3, we must select alternative #2.

Note that if we choose to implement alternative #2, our risk exposure is still $40,000. Therefore, $40,000 would be counted as a risk cost if we were calculating the cost of this project as part of a cost/benefit analysis to determine if we should initiate it.

Analyzing & Prioritizing Choices: When analyzing and prioritizing choices, measurements are utilized to rank order a set of items. We have already discussed two example of this. First, in our

problem report age discussion we established decision criteria that prioritized problem resolution priorities based on the age of the problem report. Secondly, in our discussion on cost/benefit analysis we established decision criteria that prioritized the selection between risk reduction alternatives and the resulting risk reduction leverage (RRL).

One classic example of using measurements to analyze and prioritize is to create a Pareto chart of a set of data. Pareto Analysis is the process of ranking problems or categories based on their frequency of occurrence or the size of their impact in order to determine which of many possible opportunities to pursue first. Pareto analysis is based on the Pareto Principle (80% of the trouble comes from 20% of the problems).

A Pareto Chart is a bar chart where the height of each bar indicates the frequency or impact of problems. To construct a Pareto Chart:

1. Determine the categories for the chart. For example, we might decide to examine the number of problems reported by module. Other examples might include problems by root cause, by phase introduced or by reporting customer.

2. Select a time interval for analysis. For example, we could look at all defects reported in the last 6 months.

3. Determine the total occurrences for each category. For example, the number of defects per thousand lines of code (KLOC).

4. Rank order the categories from the largest total occurrences to the smallest and create a bar chart where the bars are arranged in descending order of height from left to right.
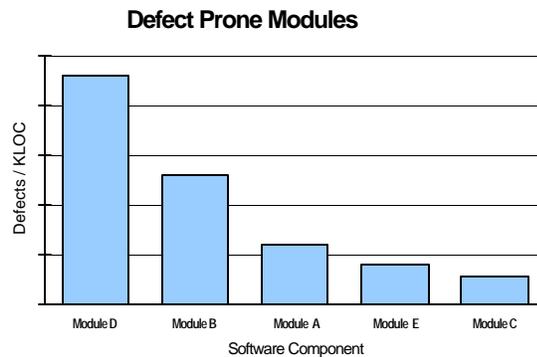
Defect Prone Modules

Figure 9: Pareto Chart - Defect Prone Modules

For example, as illustrated in Figure 9 we could create a Pareto chart of modules ranked by their defect density to prioritize those modules for reengineering or for additional defect detection effort.

Another method for analyzing and prioritizing items is to create a scorecard. Figure 10 shows an example of a scorecard for ranking and choosing suppliers. The decision criteria for this scorecard is to simply choose the supplier with the highest score.

Scorecard for Ranking Potential Suppliers

Attribute	Max Score	Supplier 1	Supplier 2	Supplier 3
Ability to deliver by date needed	10	10	7	8
Purchase price / licensing costs	10	7	5	10
Licensing restrictions	5	5	4	5
Operating costs	15	12	15	5
Maintenance costs	10	5	10	7
Process capability	10	10	8	5
Product functionality matches needs	20	18	16	8
Product quality	20	20	15	15
Ease of integration with existing systems	5	3	5	3
Ease of integration with our business processes	10	10	7	10
Ability to customize product	5	5	4	5
Technical support	5	5	3	2
Training availability	10	10	5	5
Total Score	**135**	**120**	**104**	**88**

Ability to deliver by date needed = 10 points minus one point for each week past needed date

Product functionality meets needs = (# requirements met / Total requirements) x 20

Figure 10: Supplier Selection Scorecard

Entry & Exit Criteria: Entry/exit criteria are specific, measurable conditions that must be met before the process can be started/completed. The decision criteria for one or more evaluation type metrics are frequently used as entry and/or exit criteria for a process.

For example, if we are evaluating whether or not system testing is complete, we might look at measure including:

- System Test Effort Variance decision criteria (see Figure 11): Cumulative system test effort rate matches plan within a 10% variance

- System Test Activity Status decision criteria (see Figure 12):

    o 95% of all planned test cases executed

    o No blocked test cases

    o 95% of all test cases executed were passed

- Problem report arrival rate decision criteria (see Figure 13):

    o Critical problem report arrival rate curve slope approaching zero with no new critical problems reported in last week of testing

    o Major problem report arrival rate curve slope approaching zero with no new major problems reported in last 2 days of testing
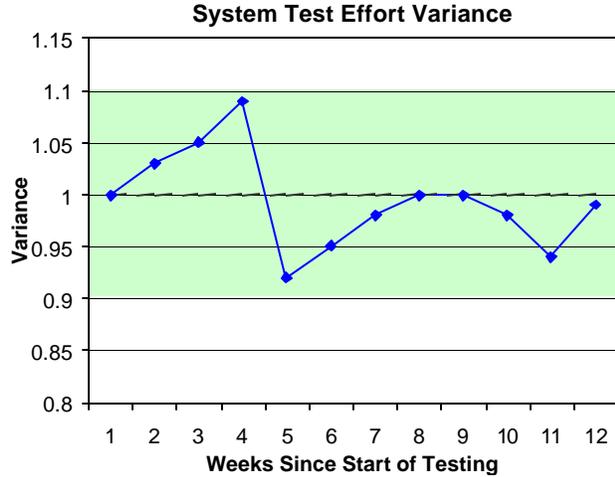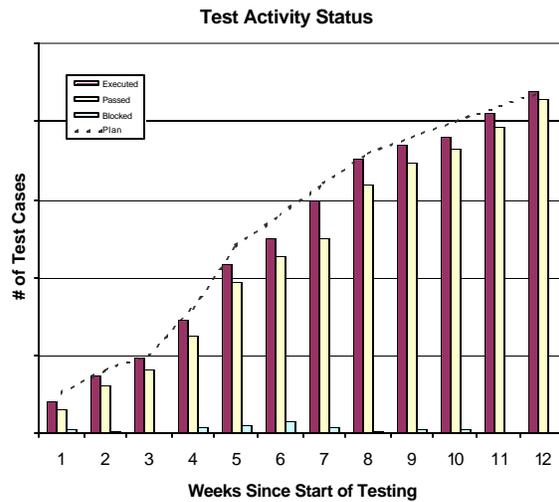
Figure 11: System Test Effort Variance

Figure 11: System Test Effort Variance

Test Activity Status

Figure 12: System Test Activity Status

Cumulative Problem Report Arrival Rates by Status
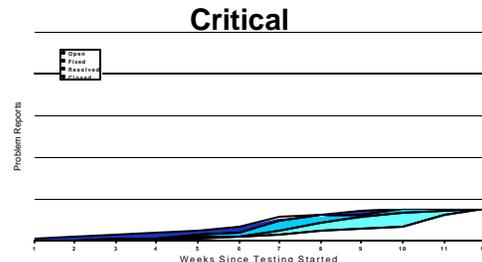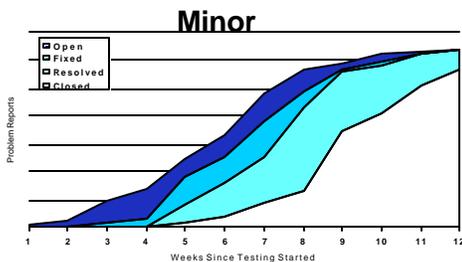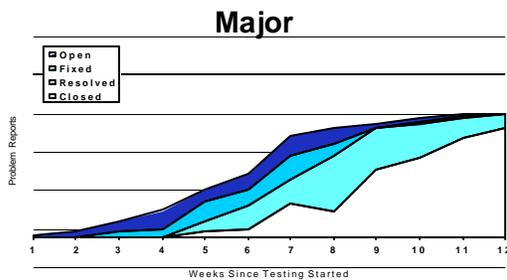
Major

Minor

Critical

Figure 13: Cumulative Problem Report Arrival Rates by Status

- o   Minor problem report cumulative arrival rate curve slope approaching zero
- Non-closed problem report counts decision criteria (see Figure 13):
    - o   Zero non-closed critical problem reports
    - o   Less than 10 non-closed major problem reports all with customer approved work-arounds
    - o   Less than 25 non-closed minor problem reports

Decision Criteria for Understand & Predict Type Metrics

Understand type metrics are used as part of research processes to learn about our software processes, products and services. Examples of using understand type metrics would be to determine:

- How much are we spending on software development? On testing?
- Where do we allocate and use resources throughout the life cycle?
- How many errors and of what types are typical in our software products? How much do they cost us?
- What are our current productivity levels?

The information from understand type metrics can be used to:

- Establish baselines, standards & goals
- Derive models of our software processes
- Examine relationships among process parameters
- Target process & product improvement efforts
- Better estimate project effort, costs & schedules

This information can then be utilized to help establish the decision used for other types of metrics. For example in our defect density discussion earlier, we had to understand the average and standard deviation for a historic set of defect density data in order to establish the thresholds used as decision criteria for controlling the defect density of the modules in our current project.

Predict type metrics are used to estimate future values of base or derived measures. Predict type metrics are used to answer questions like:

- How much will it cost? (Budget)
- How long will it take? (Schedule)
- How much effort will it take? (Staffing)
- What other resources will it take? (Resources)
- What is the probability that something will go wrong? And what will it cost if it does? (Risk)
- How many errors will it have? (Quality)
- How will those errors impact operations? (Reliability)

Going back to the ISO/IEC 15939 Software Engineering -- Software Measurement Process standard definition, decision criteria are the "thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result".   It is typically the "level of confidence in a given result" portion of this definition that applies to both understand and predict type metrics.

Considerations when determining our confidence level in a given result include:

- The completeness of the data used. For example, if we are trying to understand the amount of effort we expend on software development does our time reporting system include all the time spent including unpaid overtime?

- Is the data used subjective or objective? Has human or other types of bias crept into the data?

- What is the integrity and accuracy of the data? For example, if we again consider time card data are people completing their time cards as they complete tasks or are they waiting until the end of the week and then estimating how much time they spend on various projects.

- How stable is the product process or service being measured? For example, if we are using a new process or a new programming language, we may not be very confident in a prediction we make based on our historic data or we may not have any relevant historic data upon which to base our predictions.

- What is the variation within the data set? For example, Figure 14 shows the distribution of the responses to three different questions on a customer satisfaction survey. For each of these questions the average customer satisfaction level is 3. But how confident are we that we "understand" that the confidence level is 3 for each of these questions? If we have a distribution like the one for Question A, we can be confident that 3 represents our customer's opinions. However we would have very little confidence that a satisfaction level of 3 truly represented our customer's opinions if the response distributions looked like the one for questions B or C.
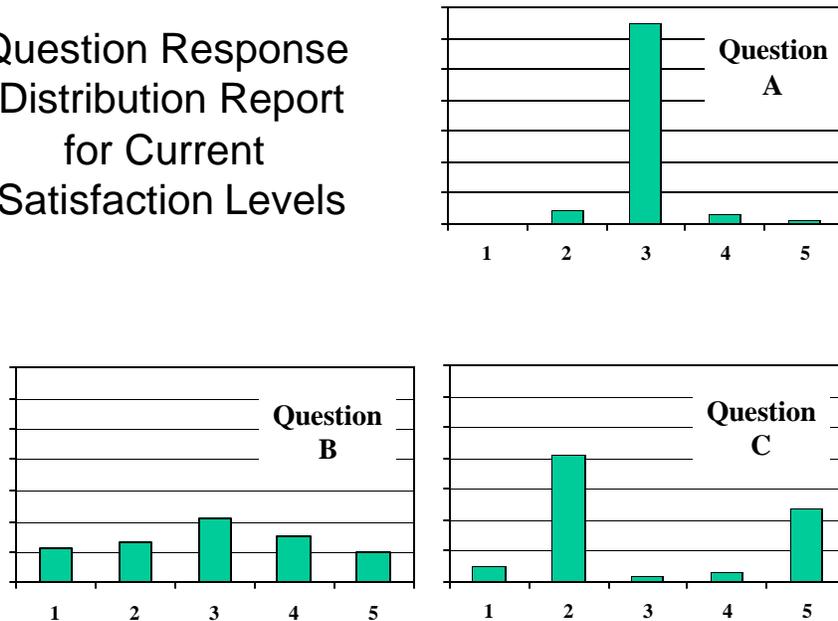

Figure 14: Customer Satisfaction Survey Results

Conclusions

I believe that having clearly defined and appropriate decision criteria are important in ensuring that our measurement indicators are correctly interpreted and applied by our measurement users. While the new ISO/IEC 15939 Software Measurement Process standard defines what decision criteria are and gives a few examples, it doesn't provide much "how to" guidance to help us establish decision criteria for our metrics. It is important to decide how we are going to use the

metrics we are planning to collect and whether we need them to control, evaluate, understand, predict, or a combination of those roles. This creates more discipline and direction in helping us determine meaningful decision criteria. There is a magnitude of ways that collected data can be compiled and presented, but how it is interpreted must be guided through careful establishment and definition of the decision criteria for each type of metric. It is my hope that the examples and discussion in this paper will provide a starting point for further discussion on how to approach what role the data is to play and how to define and establish decision criteria for different types of metrics.

References:

Humphrey-89	Watts S. Humphrey; *Managing the Software Process*; Addison-Wesley Publishing Company, Reading, MA; 1989; ISBN 0-201-18095-2.
ISO/IEC-15939:2002	International Standard, *Software Engineering – Software Measurement Process*, First edition 2002-07-15.
Jones-91	Capers Jones, *Applied Software Measurement: Assuring Productivity and Quality*, McGraw Hill, New York, 1991, ISBN 0-07-032813-7.
McGarry-01	John McGarry, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Joseph Dean and Fred Hall, *Practical Software Measurement: Objective Information for Decision Makers*, Addison-Wesley, Boston, 2001, ISBN-0-201-71516-3

Additional Resources:

Visit The Westfall Team's Software Metrics, Measures and Analytical Methods Library & Information page at:

http://www.westfallteam.com/software_metrics,_measurement_&_analytical_methods.htm

 for more papers and information including a download for a Metrics Report Definition Template.

Authors Bio & Contact Information:

Linda Westfall is president of The Westfall Team, which provides Software Metrics and Software Quality Engineering consulting and training services. Prior to starting her own business, Linda was the Senior Manager of Quality Metrics and Analysis at DSC Communications where her team designed and implemented a corporate wide metric program. Linda has over twenty years of experience in real time software engineering, quality and metrics.

Very active professionally, Linda Westfall is past chair of the American Society for Quality (ASQ) Software Division. Linda is an ASQ Certified Software Quality Engineer (CSQE) and Certified Quality Auditor (CQA) and is a Professional Engineering (PE) in Software Engineering in the state of Texas.

Linda Westfall

Address: 3000 Custer Road, Suite 270, PMB 383, Plano, TX 75075-4499

phone: 972-867-1172

email: lwestfall@westfallteam.com

web site: www.westfallteam.com