"Powerful and Pitiful Measures of Software Metrics"

Tom Gilb

Result Planning Ltd.
Powerful and Pitiful Measures of Software Metrics

by Tom Gilb (Gilb@ACM.org)

Version 21 July 99
Additions: Solingen Book Ref.

What are 'Powerful and Pitiful Measures'?
Powerful measures help management achieve their purposes as managers of software engineering teams. Powerful measures are not indirect: they measure as close to the customers' experience and need as possible. They help management to attain their customer-related objectives and to control their product development and production processes.

Measures become pitiful when they are not well suited to control what really matters. Often, inappropriate use of measures simply occurs because their use is either conventional or convenient.

To give some examples:

• function points: When used as a language-neutral measure of logic volume for comparison purposes, function points are a powerful measure. But when they are used as a 'primary or sole indicator' for estimating costs for a project, they become a pitiful measure. For any project, numerous cost drivers have to be taken into account (not least, the required quality levels).

• complexity metrics: As primary indicators of 'maintainability' or 'reliability', complexity metrics (such as McCabe's) are pitiful. They might well be extremely convenient to use (as automatic software tools are available to carry out the measuring), but they are much too indirect. Mean Time To Repair (MTTR) and Mean Time Between Failure (MTBF) are more worthwhile measures.

• defect density: Defect density is suited as a measure for controlling software process improvement: it can be used to track the injection of defects in a software process. However, using it to measure software quality is pitiful because it does not reflect 'availability' (net uptime to defined users). A reliability measure, such as MTBF in customer operational conditions, would be much more appropriate (as availability is a function of reliability (failure rate) and maintainability (fixing speed)).

Planguage
I have developed 'Planguage' for specifying requirements. In this paper, I want to show how use of measures is central to Planguage and to give some practical advice about use of measures. First, here is an outline sketch of the Planguage methods:
(Further details can be found on my web site, www.result-planning.com)
• **Specification Language** (SL); used to state system requirements. It insists on measures being identified and on quantified quality levels being specified.

• **Impact Estimation** (IE); an analysis tool (a table) allowing evaluation of the likelihood of achieving requirements and, the evaluation and comparison of different designs (strategies). A strength of IE is that it also helps identify new designs and uncover previously unstated requirements. Key measurements of IE include credibility-corrected Quality-to-Cost Ratios. Note, a useful extension of IE is to use IE tables for planning evolutionary steps and for assessing feedback.

• **Evolutionary Delivery** (Evo); based on the feedback cycles taught by the quality gurus Deming and Juran, a way of working that focuses on evolutionary delivery of early, measurable, system benefits to the customers. A system is developed, by small risk steps, in a series of plan, develop, deliver and evaluate cycles. Evo typically means that live systems are delivered step by step to user communities for trial often (e.g. weekly) and early (e.g. 2nd week of project).

• **Inspection**; a technique for measuring and improving the quality of technical specification. Technical documents are evaluated against their source documents and any prevailing standards by Inspection teams consisting of individuals with specially assigned roles. The overall aims are to identify patterns in the introduction of defects (leading to process improvement), to help train individuals to avoid creating defects, to identify defects and, to assist team-building.

In this paper, I'll discuss mainly the use of measures within the Specification Language and Impact Estimation.

**Specification Language (SL)**

SL demands specification of numerically quantified requirements. Here, I shall concentrate on specifying objectives and measures. Using SL, we must go through the following steps to specify an objective:

• Identify the objectives: *all critical* quality and resource *attributes* of the system. In practice, this could be ten or more critical qualities (e.g. availability) and, five or more critical resources (e.g. operational costs). (See later, for further discussion on identifying objectives.)

• For each attribute, specify at least one scale of measure, e.g. 'Scale: Probability of being fully operational during the office day' and 'Scale: Total of all monetary operational expenses including long term decommissioning costs'. The Scale is key to the specification as it provides the scale or units in which the measurements are specified. (See later for further discussion on selecting Scales.)

• For each Scale, define at least one Meter. A Meter is a practical means of carrying out the measurement, testing, and finding out the numeric value on the Scale.

Gilb, Powerful and Pitiful Measures
For each attribute, define one or more critical points on the defined scale of measure which are needed for the system to function properly and profitably. There are two important categories: ‘Must’ and ‘Plan’. A ‘Must’ level defines the system survival level. A ‘Plan’ level defines the planned point for success. For risk management, ‘Must’ is the first level and ‘Plan’ is the second level for risk determination. A value for any attribute less than its required Must level means total system failure. Only when all Plan levels for all the attributes have been met can a system be declared a success.

For all the Must and Plan levels, define additional qualifying information. We call this using ‘qualifiers’. You are basically defining time, place and event, i.e. when it is critical for you to achieve a certain level of an attribute, where it is critical and under what conditions. For example:

| Plan [1999, Europe, IF European Monetary Union implemented anywhere] 99.98% |
We can even give direct expression to the amount of risk we are prepared to take by a statement such as:

| Must [2001, UK, IF Euro is used in Norway & UK] 60% ±20% |
In other words the range of results 40% to 80% is an acceptable upper and lower limit, but below 40% is unacceptable.

See Figure 1 for a summary.

**Figure 1:** Specifying Objectives using Planguage. *Note: Recognize how closely this means the measures are linked to the objectives. The measures are essential to defining the objective’s failure and success criteria.*

| Objective: |
| Name: the attribute (objective) name. |
| Tag: a short identifier. |
| Gist: a brief description capturing the essential meaning. |
| Scale: a measure for the objective that is used for expressing the quality levels. |
| Meter: a practical method to be used to obtain the measurements. |
| Past: a quality or cost level (measurement) for the objective applying at a stated date in the past. |
| Must: a quality or cost level for the objective that must be achieved by the stated date or the project fails completely (i.e. level for total project survival). |
| Plan: a quality or cost level for the objective that is planned to be achieved by the stated date (i.e. success level for this one objective). |

There can be numerous Scale, Meter, Past, Must and Plan parameters for any single objective. These state the different measures and varying quality levels over time.

Gilb, Powerful and Pitiful Measures
Figure 2: A Simple Example of Planguage Specification.

<table>
<thead>
<tr>
<th>Usability:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale: Mean time to learn [defined tasks] to minimum proficiency.</td>
</tr>
<tr>
<td>Must [Release 2.0, English Version, Task: Modifying Files] 10 minutes.</td>
</tr>
</tbody>
</table>

At this point, I’ll give some further pointers about objectives and scales of measure.

**Identifying objectives:** Objectives must determine your measures. Before deciding on which measures you are going to use you must always identify your objectives. You need to recognize the true scope of your project or organization’s objectives. Objectives will be generated from a variety of directions; there will be the organization’s quality directives, the project’s business objectives, the project team’s technical objectives, the career development objectives and, no doubt, others. It may well be that the objectives that you are to monitor with your measures are of more limited scope. Fine, but do ensure that you state this and that you identify who is responsible for the other areas.

Note, there will usually be a hierarchy of objectives; one level’s strategy becoming the next level’s objectives. Keeney identifies the following levels:

Fundamental Objectives (above your ‘station’, e.g. corporate survival, reserved for the Chairman); Strategic Objectives (your concern) and ‘Means Objectives’ (which only serve to support the Strategic Objectives). This three-level objectives concept can be used in any situation, and it helps us sort out objectives we need to be concerned with, as opposed to those we cannot directly influence or be responsible for – no matter how vital they are for somebody (i.e. Fundamental Objectives). It helps us also realize that the Means Objectives, no matter how technically intriguing, are not vital to us and can be altered, replaced or dropped if we have other sufficient ways to reach our Strategic Objectives.

**Selecting Scales:** To select your Scales, consider which measures best apply to each objective. You may well choose more than one measure for a given objective. You may consider several alternative measures. You may change your measure depending on how crucial the measurement data becomes to understanding progress. The key point is that you always measure to understand your control of the objectives.

Figure 3: An example of a Scale.

<table>
<thead>
<tr>
<th>Scale: the duration in [defined time units] of [defined type] of activity done by [defined people].</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Productivity:</strong></td>
</tr>
<tr>
<td>Scale: the [Time Units, Default Units: Hours] needed to do a [Task] by [Employee Type].</td>
</tr>
<tr>
<td>Past [Time Units = Months, Task = Complaint Handling, Employee Type = Supervisor] 6 Months.</td>
</tr>
<tr>
<td>Plan [Hours, Helpdesk, Experienced, End Next Year, Europe] 60 Hours.</td>
</tr>
</tbody>
</table>
**Definition** of defined units of measure (Scale) of FLEXIBILITY:

**Scale:** relative *Speed of Response* to market opportunities and competitive conditions compared to our major competitors. "1 is same, 2 = twice as fast, 0.5 = half as fast."

**Criteria for Measures:** Ideally, you will have a collection of past experience data on Scales. The data you need one might to consider and collect on a Scale, and on its associated Meter(s) includes:

- The description of the Scale of measure and its units.
- Business relevance; what business objectives is this Scale useful for measuring?
- Technical relevance; what technical objectives is this Scale useful for measuring?
- Evidence/Source/Credibility; what is the track record of this Scale within industry, within our experience, etc.?
- The description of the Meter, i.e. the practical method of taking measurements. Note, there can be more than one Meter for each Scale.
- Uncertainty; how accurate is this Meter?
- Range; Over what range of scale does the Meter measure?
- Cost; what are the financial implications of using this Meter?
- Adaptability; Does this Meter support sampling—Can it be switched off and on—When does it deliver information—Does it need completeness—Can it target areas of greatest need—Can it cover all areas?

I take a holistic view of measures; you need to consider all types of measures. What measures are the Testing people using? What measures are the Marketing people using? What measures are your senior management using? What measures are your customers using? What are the industry standard measures? Do you need to tailor some bespoke measures for this project?

**Impact Estimation (IE)**

Now let me move on and show you how I use these quantified specifications within IE. The basic IE idea is simple: estimate quantitatively how much your design ideas impact all critical requirements. This is achieved by completing an IE table. The left-hand column of the table should contain the objectives and, across the top of the table should be the proposed strategies. For the objectives, assuming you have expressed them using

Gilb, Powerful and Pitiful Measures
Planguage, it is a question of listing down all the quality and resource attributes Tags you wish to consider. You need next to decide on a future date you want to use. This should be a system ‘milestone’; a date for which you have specified Must and Plan levels. Then, against each attribute, you state-copy from the detailed attribute definition the current level (Past) and the Plan level for your chosen date. (If you are especially risk averse you would use the Must level!) For the strategies, you simply list them by their Tag (a cross reference to a proper detailed definition of the strategies) across the top of the IE table.

You then fill in the table, for each cell you answer the question, ‘How does this strategy move the attribute from its current level towards the Plan level?’ First you state determine the actual value you would expect on the defined Scale of measure, and then. Then you convert this into a percentage of the ‘amount of required change’ (the change in level from Past to Plan). 50% means you estimate that this strategy will get you halfway to your goal on time.

For example, Training Time for Task A is currently 15 minutes and you require it to be 10 minutes within six months. You estimate Strategy B will reduce Training Time for Task A to 12 minutes. In other words, Strategy B will get you 60% of the way to meeting your objective. See Table 1.

**TABLE 1**

<table>
<thead>
<tr>
<th></th>
<th>Strategy B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Impact</td>
<td>% Impact</td>
</tr>
<tr>
<td>Training Time</td>
<td></td>
</tr>
<tr>
<td>Past = 15 minutes in June 1998</td>
<td></td>
</tr>
<tr>
<td>Plan = 10 minutes by end of Dec. 1998</td>
<td>12 minutes</td>
</tr>
<tr>
<td>Resource = Development Budget</td>
<td>$1,000</td>
</tr>
<tr>
<td>Plan = $2000 up to end Dec. 1998</td>
<td></td>
</tr>
</tbody>
</table>

There are a number of improvements to this basic idea, which make it more communicative and credible. Here is a brief summary of them:

**Uncertainty of Impact:** you can specify a range of values rather than a single value.

**Evidence for Impact Assertion:** you can state the basis for making your estimate. For example: "Strategy B was used for 5 projects last year in our company, and the percentage improvement for Training Times was always 60% to 80%".

**Source of Evidence for Impact Assertion:** Of course, some skeptic might like to check your assertion and evidence out, so you should give them a source reference, e.g. "Company Research Report ABR-017, pages 23-24."

Gilb, Powerful and Pitiful Measures
Credibility Rating of the Impact Assertion: We have found it very useful to establish a numeric 'credibility' for an estimate, based on the credibility of the evidence and the source. We use a scale of 0.0 to 1.0 (because it can then be used later to modify estimates in a conservative direction). See Table 2.

<table>
<thead>
<tr>
<th>Credibility Rating</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>wild guess, no credibility</td>
</tr>
<tr>
<td>0.1</td>
<td>we know it has been done somewhere</td>
</tr>
<tr>
<td>0.2</td>
<td>we have one measurement somewhere</td>
</tr>
<tr>
<td>0.3</td>
<td>there are several measurements in the estimated range</td>
</tr>
<tr>
<td>0.4</td>
<td>the measurements are relevant to our case</td>
</tr>
<tr>
<td>0.5</td>
<td>the method of measurement is considered reliable</td>
</tr>
<tr>
<td>0.6</td>
<td>we have used the method in-house</td>
</tr>
<tr>
<td>0.7</td>
<td>we have reliable measurements in-house</td>
</tr>
<tr>
<td>0.8</td>
<td>reliable in-house measurements correlate to independent external</td>
</tr>
<tr>
<td></td>
<td>measurements</td>
</tr>
<tr>
<td>0.9</td>
<td>we have used the idea on this project and measured it</td>
</tr>
<tr>
<td>1.0</td>
<td>perfect credibility, we have rock solid, contract-guaranteed, long-term,</td>
</tr>
<tr>
<td></td>
<td>credible experience with this idea on this project and, the results</td>
</tr>
<tr>
<td></td>
<td>are unlikely to disappear</td>
</tr>
</tbody>
</table>

Once you have completed filling in all the impacts, there are a number of calculations, using the percentage impact estimates (%Impact), that help you understand the risks involved with your proposed solution. Let me stress that these are only rough, practical calculations. Adding impacts of different independent estimates for different strategies, which are part of the same overall architecture, is dubious in terms of accuracy. But, as long as this is understood, you will find them very powerful when considering such matters as whether a specific quality goal is likely to be met or which is the most effective strategy. The insights gained are frequently of use in generating new strategies.

Impact on a Quality: For each individual quality or resource attribute, sum all the percentage impacts for the different strategies. This gives us an understanding of whether we are likely to make the planned level for each quality or cost. Very small quality impact sums like '4%' indicate high risk that the architecture is probably not capable of meeting the goals. Large numbers like 400% indicate that we might have enough design, or even a 'safety margin'.

Gilb, Powerful and Pitiful Measures
TABLE 3: Adding the percentage impacts for a set of strategies on a single quality or cost can give some impression of how the strategies are contributing overall to the objectives. Note Strategies A, B and C are independent and complementary.

<table>
<thead>
<tr>
<th></th>
<th>Strategy A</th>
<th>Strategy B</th>
<th>Strategy C</th>
<th>Sum of Strategy Impacts</th>
<th>Sum Uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>0+/-10%</td>
<td>10+/-20%</td>
<td>50+/-40%</td>
<td>60%</td>
<td>+/-70%</td>
</tr>
<tr>
<td>900-&gt;1000 hours MTBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Impact of a Strategy: For each individual strategy, sum all the percentage impacts it achieves across all the qualities to get an estimate of its overall effectiveness in delivering the qualities. The resulting estimates can be used to help select amongst the strategies. It is a case of selecting the strategy with the highest estimate value and the best fit across all the critical quality requirements. If the design ideas are complementary then the aim is to choose which strategies to implement first. If the strategies are alternatives, then you are simply looking to determine which one to pick.

TABLE 4: A measure of the effectiveness of strategy or design idea ‘Y’ can be found by adding together its percentage impacts across all the qualities

<table>
<thead>
<tr>
<th>QUALITY</th>
<th>PAST-PLAN</th>
<th>Design Idea ‘Y’</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>900-&gt;1,000 hours MTBF</td>
<td>50%+/-10%</td>
</tr>
<tr>
<td>Maintainability</td>
<td>10 min. fix to 5 min. to fix.</td>
<td>100%+/-50%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>150%+/-60%</td>
</tr>
</tbody>
</table>

In addition to looking at the effectiveness of the individual strategies in impacting the qualities, the cost of the individual strategies also needs to be considered. See next section.

Quality-to-Cost Ratio: For each individual strategy, calculate the quality-to-cost ratio (also known as the benefit-to-cost ratio). For quality, use the estimate calculated in the previous section. For cost, use the percentage drain on the overall budget of the strategy or use the actual cost.

Gilb, Powerful and Pitiful Measures
The overall cost figure used should take into account both the cost of developing or acquiring the strategy and, the cost of operationally running the strategy over the chosen time scale. Sometimes, specific aspects of resource utilization also need to be taken into account. For example, maybe staff utilization is a critical factor and therefore a strategy that doesn’t utilize scarce programming skills becomes much more attractive. My experience is that comparison of the 'bang for the buck' of strategies often wakes people up dramatically to ideas they have previously under- or over-valued.

**Average Credibility / Risk Analysis:** Once we have all the credibility data (i.e. the credibility’s for all the estimates of the impacts of all the strategies on all the qualities), we can calculate the average credibility of each strategy and, the average credibility of achieving each quality. This information is very powerful, because it helps us understand the risk involved. For example, “the average credibility, quality controlled, for this alternative strategy is 0.8”. Sounds good! This approach also saves executive meeting time for those who hold the purse strings.

**Use of IE for Evolutionary Delivery (Evo)**

IE is also of use in helping to plan the sequencing of Evo steps. IE tables can also provide a suitable format for presenting the results of Evo steps [GILB98; GILB99; MAY96]. See Table 5.

**TABLE 5:** Table 5 is a hypothetical example of how an evolutionary project can be planned and controlled and risks understood. The ‘deviation’ between what you planned and what you actually measured in practice is a good indicator of risk. The larger the deviation, the less you were able to correctly predict about even a small step. Consequently there is a direct measure of areas of risk in the ‘deviation’ numbers.

<table>
<thead>
<tr>
<th>Step→ Attribute</th>
<th>STEP1 plan %</th>
<th>actual %</th>
<th>Deviation %</th>
<th>STEP2 to STEP20 plan</th>
<th>plan cumulated to here</th>
<th>STEP21 (CA, NV, WA) plan</th>
<th>plan cumulated to here</th>
<th>STEP22 [all others] plan</th>
<th>plan cumulated to here</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUAL-1</td>
<td>5</td>
<td>3</td>
<td>-2</td>
<td>40</td>
<td>43</td>
<td>40</td>
<td>83</td>
<td>-20</td>
<td>63</td>
</tr>
<tr>
<td>QUAL-2</td>
<td>10</td>
<td>12</td>
<td>+2</td>
<td>50</td>
<td>62</td>
<td>30</td>
<td>92</td>
<td>60</td>
<td>152</td>
</tr>
<tr>
<td>QUAL-3</td>
<td>20</td>
<td>13</td>
<td>-7</td>
<td>20</td>
<td>33</td>
<td>20</td>
<td>53</td>
<td>30</td>
<td>83</td>
</tr>
<tr>
<td>COST-A</td>
<td>1</td>
<td>3</td>
<td>+2</td>
<td>25</td>
<td>28</td>
<td>10</td>
<td>38</td>
<td>20</td>
<td>58</td>
</tr>
<tr>
<td>COST-B</td>
<td>4</td>
<td>6</td>
<td>+2</td>
<td>38</td>
<td>44</td>
<td>0</td>
<td>44</td>
<td>5</td>
<td>49</td>
</tr>
</tbody>
</table>

**Further Reading**

For further reading on use of Measures, I would recommend the following:

**Raytheon:** Raytheon have published studies of their continuous improvement work (DION93, DION95). Their measures of Rework Costs, Return on Investment and Project Predictability are particularly interesting. However, their use of Defect Density (see earlier discussion) could be improved.

Gilb, Powerful and Pitiful Measures
Santa Teresa Labs: The IBM Santa Teresa Labs. book "Secrets of Software Quality" [KAPLAN94] lists a set of measures for long-term software process measurement in multiple dimensions. One innovation is that they used the scoring mechanism of the Malcolm Baldridge Award. This assesses a company on a very broad range of management and process capability. It is much broader in scope than the other measures, e.g. SEI CMM and ISO, that are also used and discussed in this book.

12 Tough Questions for Managers

I shall end by giving you my list of 'Twelve Tough Questions'. They are the "common sense" questions we need to ask on a regular basis about proposals, plans, contracts, bids, policies and strategies. They include the concepts:

- measurable results, especially in quality and benefits
- multiple critical success factor control
- estimation of results from strategies and products
- risk control through incremental result delivery.
- quality control of the plans and contracts themselves

Most of these questions can be answered with reference to ‘powerful measures’. Hopefully, this paper has demonstrated that measures can and must be used to control systems throughout their entire life-cycle.

TABLE 7: Twelve Tough Questions

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Why isn't the improvement quantified?</td>
</tr>
<tr>
<td>2.</td>
<td>What is degree of the risk or uncertainty and why?</td>
</tr>
<tr>
<td>3.</td>
<td>Are you sure? If not, why not?</td>
</tr>
<tr>
<td>4.</td>
<td>Where did you get that from? How can I check it out?</td>
</tr>
<tr>
<td>5.</td>
<td>How does your idea affect my goals, measurably?</td>
</tr>
<tr>
<td>6.</td>
<td>Did we forget anything critical to survival?</td>
</tr>
<tr>
<td>7.</td>
<td>How do you know it works that way? Did it before?</td>
</tr>
<tr>
<td>8.</td>
<td>Have we got a complete solution? Are all objectives satisfied?</td>
</tr>
<tr>
<td>9.</td>
<td>Are we planning to do the 'profitable things' first?</td>
</tr>
<tr>
<td>10.</td>
<td>Who is responsible for failure or success?</td>
</tr>
<tr>
<td>11.</td>
<td>How can we be sure the plan is working, during the project, early?</td>
</tr>
<tr>
<td>12.</td>
<td>Is it 'no cure, no pay' in a contract? Why not?</td>
</tr>
</tbody>
</table>

References


Gilb, Powerful and Pitiful Measures


**GILB98GILB99**: Tom Gilb, Various papers and manuscripts on http://www.Result-Planning.com/. The manuscripts include:

- ‘Requirements-Driven Management using Planguage’ (1995-6)
- ‘Evolutionary Project Management’ (1997)


HP Journal is available on World Wide Web at “http://www.hp.com/hpj/journal.html". (Warning HP may have moved this site, but you can get to new site from here)


**COMMENT**: This book is an excellent practical guide to useful software engineering measurementprogrammes.

UK, www.mcgraw-hill.co.uk/vansolingen/.

Author Emails:
vansolingen@bladel.rps.slb.com
e.w.berghout@its.tudelft.nl

Hardcover, 199 pages, ISBN 007 709553 7, April 1999, £34.99,
"Powerful Measures and Pitiful Measures of Software Engineering"

- IFPUG New Orleans, Oct 99,
- ©By Tom Gilb, Consultant
- Gilb@acm.org
- URL: www.Result-Planning.com
  - Where many papers, books and slides are free for your further illumination!

---

"Software Engineering" defined

- **Software engineering is**
  - the use of engineering **principles**
  - to find **designs**
  - which meet specified **requirements**
  - under conditions of **uncertainty**.
  - 'which meet' implies action until success
  - 'designs' are anything which meets requirements
  - 'requirements' are any objective, constraint, cost, time notion
  - 'uncertainty' means no off-the-shelf product, but combinations of things with complex results
Software Engineering Requirements

Design Requirements

Software Engineering Design fitted to requirements
Software Engineering under 'conditions of uncertainty'

Design

Requirements

Stages of maturity: SEI Capability Maturity Model

SEI CMM Levels: A Pitiful Measure
But: 'Management Understands it' <-- WH

<table>
<thead>
<tr>
<th>Maturity Level</th>
<th>Characteristics</th>
<th>Software Inspection Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial</td>
<td>Depends entirely on individuals.</td>
<td>none</td>
</tr>
<tr>
<td>2 Repeatabl</td>
<td>Policies, procedures, experience base</td>
<td>Writing-Task Rules, QA Policies, Inspection Procedures</td>
</tr>
<tr>
<td>3 Defined</td>
<td>Defined processes, peer reviews</td>
<td>Defect removal, Entry, Exit</td>
</tr>
<tr>
<td>4 Managed</td>
<td>Quantitative goals for product &amp; process</td>
<td>Optimum rates, quality level at exit &amp; entry, data summary, d-base</td>
</tr>
<tr>
<td>5 Optimizing</td>
<td>Entire organization. focused on continuous process improvement</td>
<td>Defect Prevention Process Improvements logging, Owners, Proc., Change Mgt., Team</td>
</tr>
<tr>
<td>6 Design</td>
<td>Design to quality &amp; cost none) Not SEI level yet.</td>
<td></td>
</tr>
</tbody>
</table>

Why Baldrige?

- Widely Used
  - N.I.S.T, Motorola, Xerox
- Proven Track Record
  - GAO Study
- Comprehensive
  - Greater scope than ISO or SEI's CMM
- Focus on Customer Satisfaction & Results

The next few slides about ISO/Baldrige are courtesy of Craig Kaplan, Ph.D.

http:\\www.iqco.com

Mapping ISO 9001 to Baldrige

<table>
<thead>
<tr>
<th>ANSI/ASQC Q91-1987</th>
<th>1994 Baldrige Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>1.0</td>
</tr>
<tr>
<td>4.2</td>
<td>2.0</td>
</tr>
<tr>
<td>4.3</td>
<td>3.0</td>
</tr>
<tr>
<td>4.4</td>
<td>4.0</td>
</tr>
<tr>
<td>4.5</td>
<td>5.0</td>
</tr>
<tr>
<td>4.6</td>
<td>6.0</td>
</tr>
<tr>
<td>4.7</td>
<td>7.0</td>
</tr>
</tbody>
</table>
Software Process Improvement at Raytheon: Some Really Useful Measures

- Source: Raytheon Report 1995: Dion et al
  - http://www.sei.cmu.edu/products/publications/95.reports/95.tr.017.html

- An excellent example of process improvement driven by measurement of improvement

- Main Motor:
  - "Document Inspection", Defect Detection

- Main Driver:
  - "Defect Prevention Process" (DPP)
**Rework Cost:**
How you can set an objective to improve it

- **Gist:** reduce by-half wasted development effort due to avoidable errors, if process improved.
- **Scale:** % of total effort which is applied to handling (identifying, correcting, re-testing, reissuing) avoidable errors.
- **Past [Our test process, 1997] 45%**
- **Plan [Us, 1998 end] 30%**,
  - [End 1999] 20%,
  - [End 2000] 10%,
  - [End 2001] 5%
Project Cost

Cost of Quality

Cost of Conformance

Cost of Performance

Cost of Non-Conformance

see next slide

Appraisal Costs

Prevention Costs

Reviews, Inspections, Testing 1st time, IV&V (1st), Audits

Training, Methodologies, Policy & Procedures, Planning, Quality Improvement Projects, Data Gathering and Analysis, Fault Analysis, Root Cause Analysis, Quality Reporting.

Costs of Non-conformance Items

- Re-reviews
- Re-tests
- Fixing Defects (code, documentation)
- Reworking any document.
- Engineering Changes
- Lab Equipment Costs of Retests

- Updating Source Code
- Patches to Internal Code
- Patches to Delivered Code
- External Failures
- from Crosby’s Model according to Raytheon95 Fig. 7

399
• $7.70 per $1 invested at Raytheon
• Sell your improvement program to top management on this basis
• Set a concrete target for it

–PLAN [Our Division, 2 years hence] 8 to 1

Raytheon 95 Software Productivity 2.7X better

Productivity

<table>
<thead>
<tr>
<th></th>
<th>Productivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1988</td>
<td></td>
</tr>
<tr>
<td>1994</td>
<td></td>
</tr>
</tbody>
</table>

170%
Achieving Project Predictability: Raytheon 95

Cost At Completion / Budget %

140%

100%

1988 1990 1994

Overall 'Product Quality':
Definition at Raytheon

- 'Overall Product Quality' = bug density!!
- The primary measure used to assess 'overall product quality' is the defect density in the final software products.
- We measure this factor in "number of software trouble reports (STRs) per thousand lines of delivered source code (STRs/KDSI)" on an individual project basis.
- The project defect densities are then combined to compute the monthly weighted average.
- data shows an improvement from an average of 17.2 STRs/KDSI to the current level of 4.0.
Overall Product Quality: Raytheon 95
Defect Density Versus Time

Examples of Process Improvements: Raytheon 95

- Process Improvements Made
  - Erroneous interfaces during integration and test
    - Increased the detail required for interface design during the requirements analysis phase and preliminary design phase
    - Increased thoroughness of inspections of interface specifications
  - Lack of regression test repeatability
  - Automated testing - Standardized the tool set for automated testing
    - Increased frequency of regression testing

- Inconsistent inspection process
  - Established control limits that are monitored by project teams
    - Trained project teams in the use of statistical process control
      - Continually analyze the inspection data for trends at the organization level

- Late requirements updates
  - Improved the tool set for maintaining requirements traceability
    - Confirm the requirements mapping at each process phase

- Unplanned growth of functionality during Requirements Analysis
  - Improved the monitoring of the evolving specifications against the customer baseline
    - Continually map the requirements to the functional proposal baseline to identify changes in addition to the passive monitoring of code growth
  - Improved requirements, design, cost, and schedule tradeoffs to reduce impacts
Fault Density versus Checking Rate: Raytheon 95

Defects Found/Kdsi

Too quick reviews and inspections will not find the defects early, thus creating lots of work for testers later.

This area is the ‘illusion of quality’

Thousands of Statements Checked per hour by a person

Risk Policy Summary

- **EXPLICIT RISK SPECIFICATION**
  - All managers/planners/engineers/metiers/quality assurance people shall immediately in writing, integrated in the main plan, specify any uncertainty, and any special conditions which can imaginably lead to a risk of deviation from defined target levels of system performance.

- **NUMERIC EXPECTATION SPECIFICATION**
  - The expected levels of all quality and cost attributes of the system shall be specified in a numeric way, using defined scales of measure, and at least one outline of one or more appropriate ‘Meters’ (Dust or measuring instruments for determining where we are on a scale).

- **CONDITIONS SPECIFIED**
  - The requirements levels shall be qualified with regard to when where and under which conditions the targets apply, so there is no risk of us inadvertently applying them inappropriately.

- **COMPLETE REQUIREMENT SPECIFICATION**
  - A complete set of all critical quality and cost aspects shall be specified, avoiding the risk of failing to consider a single critical attribute.

- **COMPLETE DESIGN SPECIFICATION and IMPACT ESTIMATION**
  - A complete set of designs or strategies for meeting the complete set of quality and cost targets will be specified. They will be validated against all specified quality and cost targets (using Impact Estimation Tables). They will meet a reasonable level of safety margin. They will then be evolutionarily validated in practice before major investment is made. The Evo steps will be made at a rate of maximum 2% of budget, and 2% of ‘project time’, per ‘incremental crit’ (Evo step) of design or strategy.

- **SPECIFICATION QUALITY CONTROL NUMERICALLY EXITED**
  - All requirements, design, impact estimation and Evolutionary project plan, as well as all other related critical documents such as contracts, management plans, contract modifications, marketing plans, shall be ‘quality controlled’ using the Inspection method [GBBS]. A normal process Exit level shall be that ‘no more than 0.2 Major Defects per page maximum, can be calculated to remain, as a function of those found and fixed before release, when checking is done properly’ (e.g. at optimum checking rates of 1 logical page or less per hour).

7. **EVOLUTIONARY PROOF-OF-CONCEPT PRIORITIES**

- The Evolutionary Project Management method [GGBS, GGBS] will be used to sense and control risk in mid-project. The dominant paradigms will be:
  
  - **Get early sunk costs into sunk cost status.**
  - **Get out early sunk costs and re-allocate (X) per function per check out of sunk cost deferral in the Evolutionary process, or out to vendor, or with ‘vendor funded’ in support of project budget.”
1. EXPLICIT RISK SPECIFICATION

- All
  - managers/planners/engineers/testers/ quality assurance people shall
  - immediately in writing, integrated in the main plan,
  - specify any uncertainty,
  - and any special conditions
    - which can imaginably lead to a risk of deviation
    - from defined target levels of system performance.
2. NUMERIC EXPECTATION SPECIFICATION

- The expected levels of all quality and cost attributes of the system shall be
  - specified in a numeric way, using defined Scales of measure, - - - - -
  - and at least an outline of one or more appropriate ‘Meters’
    - test or measuring instruments for determining where we are on a scale.

3. CONDITIONS SPECIFIED

- The requirements levels shall be [qualified]
  - with regard to
    - [when]
    - [where]
    - and under which [conditions] the target requirements apply,
    - so we don’t shoot for the wrong target!
4. COMPLETE REQUIREMENT SPECIFICATION

• A complete set of
  – all critical quality and cost aspects
  – shall be specified,
  – avoiding the risk of failing to consider even one critical attribute.

5. COMPLETE DESIGN SPECIFICATION and IMPACT ESTIMATION:

• A complete set of designs or strategies for meeting the complete set of quality and cost targets will be specified.

• They will be validated against all specified quality and cost targets (using Impact Estimation Tables).

• They will meet a reasonable level of safety margin.

• They will then be evolutionarily validated in practice before major investment is made.
  – The Evo steps will be made at a rate of maximum 2% of budget, and 2% of ‘project time’, per ‘incremental trial’ (Evo step) of designs or strategies.
Results Language Icons for requirements, design and design
goodness

How can we express the goodness of a design idea?

Quality-5

'Benchmark'

60% of

'Design Target'

(100% level)

Impact Table 3D Display:

How good are design ideas compared to your objectives?

good stuff indicator

Quality
Requirements
(defined
quantitatively)

Tracking System
Show & Tell
Project Survey
Training

Design Ideas
(specified in detail
elsewhere)
### Impact Estimation concepts: full table

<table>
<thead>
<tr>
<th>Strategies/Obectives</th>
<th>A1</th>
<th>B4</th>
<th>CD</th>
<th>DX</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability &quot;99.9%-99.98%&quot; &quot;Past-&gt;Plan&quot;</td>
<td>0%</td>
<td>100%</td>
<td>50%</td>
<td>-5%</td>
<td>145%</td>
</tr>
<tr>
<td>Portability &quot;80%-95%&quot;</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4%</td>
</tr>
<tr>
<td>Usability &quot;3 mins-&gt;1&quot;</td>
<td>60 ±20</td>
<td>99</td>
<td>41</td>
<td>200</td>
<td>400%</td>
</tr>
<tr>
<td>Budget &quot;0-&gt;1 million&quot;</td>
<td>100</td>
<td>10%</td>
<td>?</td>
<td>n.a.</td>
<td>110?</td>
</tr>
<tr>
<td>Employers &quot;0-&gt;32 people&quot;</td>
<td>0</td>
<td>30%</td>
<td>?</td>
<td>n.a.</td>
<td>9±5</td>
</tr>
<tr>
<td>Benefit/Cost-&gt;</td>
<td>0.6</td>
<td>5.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Impact Estimation concepts: detail

- **Strategies/Obectives**: 0% 100% 50% -5% 145%
- **Portability**: 1 1 1 1 4%
- **Usability**: 60 ±20 99 41 200 400%
- **Budget**: 100 10% ? n.a. 110?
- **Employers**: 0 30% ? n.a. 9±5
- **Benefit/Cost**: 0.6 5.0

**Quality and Benefit Objectives**

- **Strategies/Obectives**: A1 B4 CD DX Sum
- **Availability**: 0% 100% 50% -5% 145%
- **Portability**: 1 1 1 1 4%
- **Usability**: 60 ±20 99 41 200 400%
- **Budget**: 100 10% ? n.a. 110?
- **Employers**: 0 30% ? n.a. 9±5
- **Benefit/Cost**: 0.6 5.0

**Objective statement example**

- **SCAT**: Avg. Minutes for typical user to learn to operate our product.
- **Meter**: At least 100 users.
- **Past**: New Product: 3 minutes.
- **Plan**: New Product: 19595.1 minutes.

**Notes**

- A1: Graphical interfaces using minimal language, no codes, maximum pictures, maximum user tailoring, maximum learning about particular users.
6. SPECIFICATION QUALITY CONTROL NUMERICALLY EXITED

- **All Specifications**
  - {requirements, design, impact estimation and Evolutionary project plans,
  - as well as all other related critical documents such as contracts, management plans, contract modifications, marketing plans,}
  - shall be ‘quality controlled’ using the Inspection method.

- **A normal process Exit level shall be**
  - that ‘no more than 0.2 Major Defects per page maximum, can be calculated to remain,
  - as a function of those found and fixed before release,
  - when checking is done properly’ (e.g. at optimum checking rates of 1 logical page or less per hour).

---

**Inspections and Reviews**

- **Work Product** → **Inspection:**
  - Meet standards?
  - **Exit:**
    - Exited Document
  - **Review:**
    - Go No-Go

- **Inspections**
  - Judgement based on conformance to standards
  - Well written, clear, complete, trustworthy
  - Can be carried out by any of ‘intended readership’
  - Should be done to guarantee decision-makers a good basis for a decision.

- **(Go No-Go) Reviews**
  - Judgement based on goodness in real world
  - Content, not format;
  - Value, not clarity
  - Approval by authorized ‘managers’
  - Should not be Entered if document not Exited from Inspection
7. EVOLUTIONARY PROOF-OF-CONCEPT PRIORITIES

- The Evolutionary Project Management method will be used to sense and control risk in mid-project. The dominant paradigms will be
  - 2% steps,
  - high value to cost with regard to risk delivered first.
    » high risk strategies tested 'offline to customer delivery', in the Backroom of development process, or at cost-to-vendor, or with 'research funds' as opposed to project budget.

---

Evo Example:
Usability Goal, Complex example

- Usability:
- Gist: the relative ease of learning and using a defined product compared to previously used products.
- Scale: average minutes per [defined User type] to learn to use [defined Tasks to use the product].
- Meter: at least 30 users of representative defined User type will be monitored doing at least 10 defined Tasks of defined function type.
- Past [Old Product PP, Home Buyer, Adult, Task: Build telephone number list] 30 minutes.
- Record [MM, Adult, Task: Dial Out] 10 seconds ← Consumer Reports, January
- Wish [Our Customers, Mix] 5 minutes ← Chairman's Dream in Speech
- MinLevel: Must [Our Customers, Mix, New Product] 10 minutes ← marketing minimum
- Plan [Our customers, Mix, New Product, First Field Release] 50% of MinLevel ← Guess by Project Mgr., [within 2 years of First Field Release] 30% of MinLevel ← Guess.
  * Local Definitions of Terms.
- Mix: Defined: representative mix of common frequent user tasks.
- User: Defined: person who intends to use the product in the long term, not a test person.
- First Field Release: Defined: First sold releases to any public market after Field Trials.
Usability Specification example

Wish 5 minutes <- Chairman

Plan [1st rel.] 50% of 10 minutes
Must 10 mins.

Targets

Plan [within 2 yrs.
1st rel.] 30% of 10
minutes

Usability: -|---|--- minutes for User to do Task

Benchmarks

Product
Past 30 minutes
Trend 20 minutes
Record 10 secs

Usability Evo Delivery Steps to meet Plan

Wish 5 minutes <- Chairman

Plan [1st rel.] 50% of 10 minutes
Must 10 mins.

Targets

Plan [within 2 yrs.
1st rel.] 30% of 10
minutes

Product
Evo Step 1
Step 2
Step 3
Step 4 k

Benchmarks
Past 30 minutes
Trend 20 minutes
Record 10 secs
### Impact Table for Step Management

<table>
<thead>
<tr>
<th>Step #1 Plan</th>
<th>Step #1 Actual</th>
<th>Differe nce</th>
<th>Total Step 1</th>
<th>Step #2 Plan</th>
<th>Step #2 Actual</th>
<th>Step #2 Difference</th>
<th>Total Step 1+2</th>
<th>Step #3 Next step plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: (Design X, Function Y)</td>
<td>50% ±30%</td>
<td>49%</td>
<td>-10%</td>
<td>60%</td>
<td>100% ±20%</td>
<td>80%</td>
<td>-20%</td>
<td>120%</td>
</tr>
<tr>
<td>B: (Design Z, Design F)</td>
<td>80% ±40%</td>
<td>49%</td>
<td>-40</td>
<td>0</td>
<td>30%</td>
<td>70%</td>
<td>30%</td>
<td></td>
</tr>
<tr>
<td>C: (Design G, Design H)</td>
<td>10% ±20%</td>
<td>12%</td>
<td>+2%</td>
<td>26% ±15%</td>
<td>5%</td>
<td>-15%</td>
<td>17%</td>
<td>83%</td>
</tr>
<tr>
<td>D: (Design I, Design J)</td>
<td>20% ±1%</td>
<td>10%</td>
<td>-10%</td>
<td>10%</td>
<td>5% ±2%</td>
<td>10%</td>
<td>-5%</td>
<td>20%</td>
</tr>
<tr>
<td>E: (Design K, Design L)</td>
<td>2% ±1%</td>
<td>4%</td>
<td>-2%</td>
<td>4%</td>
<td>10% ±2.5%</td>
<td>3%</td>
<td>+7%</td>
<td>7%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Calendar Time</th>
<th>1 week</th>
<th>2 weeks</th>
<th>1 week</th>
<th>0.5 weeks</th>
<th>2.5 weeks</th>
<th>1 week</th>
</tr>
</thead>
</table>

### 10 Principles of Risk Management: Summary

- 1. Frequent Feedback
  - Early frequent and measurable feedback from reality must be planned into your development process, to identify risks before they become dangerous.
- 2. Rigorous Requirements
  - All critical success-and-failure quality/performance/cost requirements must be identified, made measurable and tracked through design and evolutionary deployment.
- 3. Requirement Impact Estimation
  - A design phase must address all critical few requirements and systematically estimate the impact of all design ideas on all critical requirements.
- 4. Upstream Pollution Control
  - All upstream documents (requirements, design) must be thoroughly inspected against a strong set of Rules for Good Practice, and not exited to next phases until they have reached a reasonable level of Major Defect Freeens.
- 5. Personal Risk Responsibility
  - People must be given personal responsibility in their sector for identification and mitigation of risks.
- 6. Design Out Risk
  - Unacceptable risk needs to be 'designed out' of the system consciously at all levels of engineering, architecture, purchasing, contracting, development process, motivation and maintenance process.
- 7. Maximum Risk Policy
  - The total level of risk exposure at any one stage should be consciously reduced to a minimum of about 2-5% of total budget, even with total failure of that stage alone.
- 8. Maximize profit, not minimize risk itself
  - Focus not on elimination of all risk, but on maximization of benefit to cost result delivery, even considering risks.
- 9. Backups are part of the Price
  - Conscious planning and development of backup for risks is a necessary minimum cost of planning and projects.
- 10. Contract Out Risk
  - Make vendors contractually responsible for risks, they will give you better advice and services as a result.
‘Pop’ Version Risk Principles 1->5 of 10

1. Frequent Feedback
   - You’ve gotta shoot real bullets to see if they shoot back

2. Rigorous Requirements
   - Your project Achilles Heel is the critical requirement that you didn’t think was so critical.

3. Requirement Impact Estimation
   - Design for success, or failure will have designs on you.

4. Upstream Pollution Control
   - If you’re infected, don’t kiss your friends hello!

5. Personal Risk Responsibility
   - Take risks personally, or they will take you personally.

‘Pop’ Version Risk Principles 6->10 of 10

6. Design Out Risk
   - If you don’t choose the risk level, risk will make it’s own decisions.

7. ‘Maximum Risk’ Policy
   - If you might fail, you might as well do it on a small scale.

8. Maximize profit, not minimize Risk itself
   - Risk is just a disturbance, profit is the point

9. Backups are part of the Price
   - If you leave the umbrella at home, that’s when it rains.

10. Contract-Out Risk
    - Let others bear the risks at hand, so they will join your merry band
10 Principles of Risk Management: which relate to Software Metrics

1. Frequent Feedback

"You've gotta shoot real bullets to see if they shoot back"

- Early frequent and measurable feedback from reality must be planned into your development process, to identify risks before they become dangerous.
2. Rigorous Requirements:

"Your project's 'Achilles Heel' is the critical requirement that you didn't think was 'so' critical."

| All critical success-and-failure quality/performance/cost requirements must be identified, made measurable and tracked through design and evolutionary deployment. |

3. Requirement Impact Estimation:

"Design for success, or failure will have designs on you."

| A design phase must address all critical few requirements and systematically estimate the impact of all design ideas on all critical requirements. |
4. Upstream Pollution Control

- If you're infected, don't kiss your friends
  hello!

- All upstream documents (requirements, design) must be thoroughly inspected against a strong set of Rules for Good Practice, and not exited to next phases until they have reached a reasonable numeric level of Major Defect Freeness.
  (like Max. 3.0 or 0.3 Maj/Pg)

5. Personal Risk Responsibility:

"Take risks personally, or they will take you personally."

- People must be given personal responsibility in their sector for identification and mitigation of risks.
6. Design Out Risk:
"If you don't choose the risk level, risk will make it's own decisions."

- Unacceptable risk needs to be 'designed out' of the system consciously at all levels of engineering, architecture, purchasing, contracting, development process, motivation and maintenance process.

7. Maximum Risk Policy:
"If you might fail, you might as well do it on a small scale."

- The total level of risk exposure at any one stage should be consciously reduced to a minimum of about 2-5% of total budget, even with total failure of that stage alone.
8. Maximize profit, not minimize Risk itself:
   "Risk is just a disturbance, profit is the point"

- Focus not on elimination of all risk,
  - but on maximization of 'benefit-to-cost' result delivery,
  - even considering risks.

9. Backups are part of the Price:
   "If you leave the umbrella at home, that's when it rains."

- Conscious planning and development of backup for risks is a necessary minimum cost of planning and projects.
10. Contract Out Risk:

"Let others bear the risks at hand, so they will join your merry band"

- Make vendors contractually responsible for risks,
- they will give you better advice and services as a result.

Last Slide
Goodbye!

Note: this is skipped to by summary of risk principles