



Impact Estimation Tables

Understanding Complex Technology Quantitatively

Tom Gilb
Independent Consultant

How good is your design suggestion? Does anybody else understand why you think the technology you suggest is such a great idea? Would you like to know how to shoot down those dumb ideas that consultants and your colleagues use to entice your managers? Would you like a great approach to prove your technical expertise to the world? We may have it right here.

Impact Estimation (IE) tables allow you to analyze any technical or organizational idea in relation to requirements and costs. It is a method I have developed over the last 20 years, and it works! To give one example, shortly after we taught the idea to a manufacturing group, they declared it was worth a million dollars. Using IE for the first time, they presented a bid for project money to management and got the full budget they requested—\$1 million more than they had expected!

Aims of IE

IE can be used for a wide variety of purposes [1]. Its most important uses include

- Comparing alternative design ideas.
- Estimating the state of the overall design architecture.
- Planning and controlling evolutionary project delivery steps.
- Analyzing risk.

I use IE tables when evaluating projects to help answer my “Twelve Tough Questions.”

- Why is the improvement not quantified?
- What is the degree of the risk or uncertainty and why?
- Are you sure? If not, why not?
- Where did you get that information? How can I check it out?
- How does your idea measurably affect my goals?
- Did we forget anything critical to survival?
- How do you know it works that way? Did it before?
- Have we got a complete solution? Are all objectives satisfied?
- Are we planning to do the “profitable things” first?
- Who is responsible for failure or success?
- How can we be sure the plan is working during the project—early?
- Is it “no cure, no pay” in a contract? Why not?

A More Quantitative Approach

The basic IE idea is simple: Estimate quantitatively how much your design ideas impact all critical requirements. As simple as this is, software engineers do not normally do it. We judge too narrowly. We only treat the costs, e.g., development costs and operational costs, quantitatively. The qualities, e.g., usability, system availability, and system flexibility, tend to be handled subjectively. There are two important underlying issues here. First, we need to express our requirements in a quantitative manner. Second, we need to gather objective data about our technologies. We can make a start by making use of practical experience data.

How to Quantify and Document the Relationship Between Requirements and Design

First, I will show how to express the relationships between the system requirements and your new design ideas using IE. Later, I will show how this analysis and presentation discipline can be used to analyze many design ideas (the overall system architecture) in relation to all system requirements.

IE focuses on the system qualities and costs. It is a question of how a design idea impacts all the critical *qualities*, e.g., performance, usability, and reliability, and how it impacts all the costs (money, time, people, and space) to build, deliver, and maintain the design idea. A design idea is “effective” to the degree it satisfies specified requirement levels of qualities. A design idea is cost-effective (“efficient”) to the degree it is effective in relation to all costs.

Note: IE does not consider functions. The quality and cost requirements for the system are the prime considerations. Design ideas are evaluated as to how “good” and cost-effective they will be at delivering the qualities required. For example, when considering how to transport a person from point A to point B, it is the quality and cost requirements (such as flexibility of travel times, safety, reliability, comfort, and price) that select the best design idea (such as airplane, foot, rocket, or ambulance).

A Simple Relationship Between Requirements and Design Ideas

I will start simple and expand scope later. Assume I have a requirement (a constraint) that my budget not exceed \$100,000. Also assume that I have a design idea—I will call it

Table 1. A simple IE table.

Design Idea: Big Idea	Real Impact	%IMPACT
Quality = Reliability Plan = 1,000 hours MTBF by end of December 1999	1,000 hours MTBF	100 percent
Cost = Development Budget Plan = \$100,000 up to end of December 1999	\$10,000	10 percent

Credibility Rating	Meaning
0.0	Wild guess, no credibility.
0.1	We know it has been done somewhere.
0.2	We have one measurement somewhere.
0.3	There are several measurements in the estimated range.
0.4	The measurements are relevant to our case.
0.5	The method of measurement is considered reliable.
0.6	We have used the method in-house.
0.7	We have reliable measurements in-house.
0.8	Reliable in-house measurements correlate to independent external measurements.
0.9	We have used the idea on this project and measured it.
1.0	Perfect credibility, we have rock solid, contract-guaranteed, long-term, credible experience with this idea on this project, and the results are unlikely to disappear.

Table 2. An example of how credibility ratings can be assigned.

Big Idea—that I estimate will cost \$10,000 (10 percent of my budget).

I also have a quality requirement—I will call it *Reliability*—which is to reach mean time between failures (MTBF) of 1,000 hours by the end of December 1999. If I believe that *Big Idea* will reach 1,000 hours MTBF within the time scales, it satisfies my requirement for *Reliability* completely (100 percent). You can express this with a simple IE table (Table 1).

Further Improvements to Specifying the Relationship

There are a number of improvements to this basic idea, which make it more communicative and

credible. Following is a brief summary of them.

Impact Relative to a Defined Baseline

For all qualities, it is essential to define a baseline. Usually, the current value achieved by the “old” system is used. For example, if “900 hours MTBF” was the level for reliability of our previous system, we could use that as a reference base. This can be stated using Planguage, my Requirements Planning Language [1], as PAST [September 1997] 900 hours MTBF. It is the minimum level or 0 percent level because only when the system’s MTBF is higher than 900 hours will any progress have been made on improving reliability.

The planned level by December 1999 is 1,000 hours MTBF. Using Planguage, this is expressed as PLAN [December 1999] 1,000 hours MTBF. This is the 100 percent level. When we have improved the MTBF to this level, we shall have completely met our reliability requirement. (Exceeding a requirement is fine as long as we have not incurred additional costs or failed to divert resources that could have been expended on achieving other requirements.)

If the impact of *Big Idea* was estimated at 900 hours, we would be making no forward progress toward our planned level. In other words, the percentage impact of *Big Idea* (from base to plan) is 0 percent. It is unlikely that anyone needs a new idea to get to where they are already.

However, if the impact of *Big Idea* was estimated at 950 hours MTBF, it is a much more interesting design idea. Its percentage impact is 50 percent, halfway between the base of 900 hours (0 percent) and the goal of 1,000 hours (100 percent).

Note, the percentage impact on plan (%IMPACT) can only be estimated if there is both a baseline and a planned

level. %IMPACT is useful because it enables us to “add” the percentage impacts from different scales-of-measure, as will be explained later.

Uncertainty of Impact

All estimates are uncertain. It is useful to estimate how uncertain they are. This helps you understand the risk of not meeting desired goals. So if the uncertainty for the estimate of 950 hours MTBF was plus or minus 10 hours, our estimate of 950 hours becomes 950+/-10, which could be expressed alternatively as a percentage impact of 50 +/-10 percent. The negative number (-10 percent) can be used to modify estimates to discover the worst-case situation.

Evidence for Impact Assertion. If you want any credibility for your assertions, you should be prepared to supply facts to back them up. Instead of waiting to be asked, dig up the facts and document them in advance. This shocks people—they are not accustomed to being offered facts. For example, “*Big Idea* was used for 10 projects last year in our company, and the range of MTBF attributed to it was 940 to 960 hours MTBF, average 950.”

Source of Evidence for Impact Assertion. Of course, some skeptics might like to verify your assertion and evidence, so you should give them a source reference, e.g., “Company Research Report TR-017, pp. 23-24.”

Credibility Rating of the Impact Assertion

We have found it extremely useful (it was a key part of getting the \$1 million mentioned earlier) 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 for an example of how credibility ratings can be assigned.

Table 3. Example: The set of data for a single-cell estimate of the impact of *Big Idea* on reliability.

Requirement: Past → Plan	Real Impact (Estimate on real scale for <i>Big Idea</i>)	Relative to plan (and base estimate)	Uncertainty Estimate	Evidence	Source	Credibility
Reliability: 900 → 1,000 hours MTBF	950	50 percent	+/- 10%	Project A 940 hours MTBF	TR-017	0.6

The credibility rating is useful because it forces you to analyze, gather data, and raise the credibility to an acceptable level. Normally, people make no effort here.

Further Analysis of the IE Data

Now assume you have numerous critical qualities and that you have completed an IE table using several design ideas. There are now a number of calculations using the %Impact estimates you can do to help understand the robustness of your proposed solution.

I stress that these are only rough, practical calculations. Adding impacts of different independent estimates for different design ideas that are part of the same overall architecture is dubious in terms of accuracy. But as long as this is understood, you will find them extremely powerful when considering such matters as whether a specific quality goal is likely to be met or which is the most effective design idea. The insights gained are frequently of use in generating new design ideas.

I add an additional cautionary note: I expect this information to only be used as a rough indicator to help designers spot potential problems or select design ideas. Any real estimation of the impact of many design ideas needs to be made by real tests; ideally, by measuring the results of early evolutionary steps in the field.

Impact on Quality

For each individual quality or cost, sum all the percentage impacts for the different design ideas. This gives us an understanding of whether you are likely to make the planned level for each quality or cost. Extremely small quality impact sums like 4 percent indicate high risk that the architecture is probably not capable of meeting the goals. Large numbers like 400 percent indicate you might have enough design or even a "safety margin."

Impact of a Design Idea

For each individual design idea, 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 among the design ideas. It is a case of selecting the design idea with the highest estimate value

Table 4. Example: Adding the percentage impacts for a set of design ideas on a single quality or cost can give some impression of how the designs are contributing overall to the project goals. Note: Design Ideas A, B, and C are independent and complementary.

Quality	Past → Plan	Big Idea
Reliability	900 → 1,000 hours MTBF	50% +/- 10%
Maintainability	10 min. to fix → 5 min. to fix	100% +/- 50%
Estimate of effect of Big Idea on all goals		150% +/- 60%
Averaged Estimate		75% +/- 30%

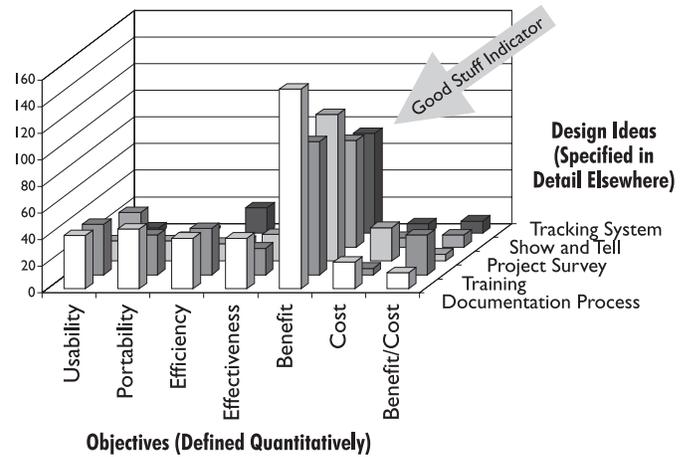


Figure 1. A 3-D representation of an IE table. "Benefit" shows the summed total of all the %Impacts for the qualities. Note: The design idea (Documentation Process) that contributes most toward meeting the requirements is shown to not be the most cost-effective. Note also, the summed total for each of the objectives, i.e., the totals if all the design ideas were implemented, has not been calculated.

and the best fit across all the critical quality requirements. If the design ideas are complementary, the aim is to choose which design idea(s) to implement first. If the design ideas are alternatives, you are merely looking to determine which one to pick.

These estimates are something like universal currency. Each estimate tells how well you are moving toward your goals. It allows you to get some impression of the overall impact of a single design idea. For example, if a design idea has 50 percent impact on each of the different goals, I might console myself by rationalizing that I have designed enough to get halfway to my goals.

In addition to looking at the effectiveness of the individual design ideas in impacting the qualities, the cost of the individual design ideas also needs to be considered, as will be shown in the next section.

Quality-to-Cost Ratio

For each individual design idea, 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 design idea or use the actual cost.

The overall cost figure used should take into account both the cost to develop or acquire the design idea and the cost of operationally running the design idea 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; therefore, a design idea that does not use scarce programming skills becomes much more attractive.

My experience is that a comparison of the impact vs. the cost of design ideas often wakes people up dramatically to ideas they have previously undervalued or overvalued.



This article can be found in its entirety on the Software Technology Support Center Web site at <http://www.stsc.hill.af.mil/CrossTalk/crostalk.html>. Go to the "Web Addition" section of the table of contents.

National Software Quality Experiment

A Lesson in Measurement: 1992-1997

Don O'Neill, *Independent Consultant*

The nation's prosperity is dependent on software. The National Software Quality Experiment is riveting attention on software product quality and revealing the patterns of neglect in the nation's software infrastructure. In 1992, the Department of Defense Software Technology Strategy set the objective to reduce software problem rates by a factor of 10 by the year 2000. The National Software Quality Experiment is being conducted to benchmark the state of software product quality and to measure progress toward the national objective. Motivation for the experiment, methods used to collect data, and an analysis of the findings are presented.

Average Credibility and Risk Analysis

Once you have all credibility data, i.e., the credibilities for all the estimates of the impacts of all the design ideas on all the qualities, you can calculate the average credibility of each design idea and the average credibility of achieving each quality. This information is powerful because it helps you understand the risk involved; for example, "the average credibility, quality controlled, for this alternative design idea is 0.8." Sounds good. This approach also saves meeting time for those who hold the purse strings.

As long as you do not get carried away and attempt too many calculations, it can help your understanding of the risks involved if you modify your estimates using the worst-case error or credibility ratings. Altering an estimate to its worst-case value is a matter of subtracting or adding the worst-case error correction to the estimate. To modify an estimate to take into account its credibility, multiply the estimate by its credibility rating. Multiplying a worst-case estimate by its credibility rating will give you the most pessimistic value. Once you have reworked the estimates, you can then recalculate the totals.

Conclusion

If you want to move software engineering toward "real" engineering and toward better control over your results, introduce systematic and fact-based thinking into software development.

	Design Idea A	Design Idea B	Design Idea C	Sum of Impacts	Sum Uncertainty
Reliability 900 → 1,000 hours MTBF	0 +/- 10%	10 +/- 20%	50 +/- 40%	60%	+/- 70%

Table 5. A measure of the effectiveness of Big Idea can be found by adding together its percentage impacts across all the qualities.

Selective use of IE tables by project management would greatly assist communication with senior management and improve risk control.

Acknowledgments

I thank Barry Boehm of the University of Southern California, who at the 1974 International Federation of Information Processing Societies Conference in Stockholm showed me his Requirements/Properties Matrix, a precursor to Quality Function Deployment, which challenged me to provide the quantification aspects of IE tables. He has always provided an appreciative audience to my evolution of the method and is an inspiration to us all. I also thank Steve McConnell for his initial suggestion, help, and comments on this article and Lindsey Brodie for editing this article. ♦

About the Author



Tom Gilb immigrated from California to Europe in 1956. In 1958, he began working for IBM in Norway, and five years later became a

freelance consultant. He is author of several books, including *Software Metrics* (1976-77), *Software Inspection* (1993), and *Principles of Software Engineering Management* (1998). He spends about half his time working in the United States and half in Europe.

Internet: <http://www.result-planning.com>
E-mail: Gilb@ACM.org

Reference

1. Gilb, Tom, "Requirements-Driven Management Using Planguage," http://www.stsc.hill.af.mil/SW_Testing/gilb.html, <http://www.result-planning.com>, 1995-1996.

Recommended Reading

1. Akao, Yoji, ed., *Quality Function Deployment: Integrating Customer Requirements into Product Design*, Productivity Press, Cambridge, Mass., 1990.
2. Gilb, Tom, *EVO: The Evolutionary Project Manager's Handbook*, <http://www.result-planning.com>.
3. Gilb, Tom, *Principles of Software Engineering Management*, Addison-Wesley Longman, Boston, Mass., 1988.