How problems with Quality Function Deployment's (QFD's) House of Quality (HoQ) can be addressed by applying some concepts of Impact Estimation (IE)

Tom Gilb
Result Planning Limited
Tom.Gilb@INCOSE.org

Lindsey Brodie
Middlesex University
L.Brodie@mdx.ac.uk

Abstract. Quality Function Deployment (QFD) is widely taught as a method for analyzing the relationship between quality and design. The structure of QFD – multiple qualities correlated to multiple designs – leads to a very healthy and necessary analysis process (Clausing 2007). However there are several problems with QFD, which need to be addressed. The main concern is the use of ordinal and ratio scales of measure. This paper proposes that Planguage's Impact Estimation (IE) has some concepts that could usefully be harnessed within QFD to address this.

Note this paper mainly discusses the House of Quality (HoQ) within the traditional QFD method, which is widely used in the US and Europe.

Introduction

This paper proposes modifications to the House of Quality (HoQ) within Quality Function Deployment (QFD). The suggestion is to replace ordinal and ratio scale data with data on an absolute scale, namely Planguage metrics as used in Planguage's Impact Estimation (IE). The benefits include more meaningful calculations, and better communication of information.

This paper will briefly describe traditional QFD as used in the IT industry in US and Europe, and discuss some of its known problems before describing IE and outlining how use of Planguage metrics could help address some of the QFD problems.

Traditional QFD

QFD originated in Japan in the mid-1960s, developed by Dr. Yoji Akao and Dr. Shigeru Mizuno, as a method for assuring the quality of new products. See Figure 1, which shows the House of Quality (HoQ) matrix, which is one part of QFD. In practice, the majority of users stop after they have completed the HoQ (Cohen 1995), but QFD is meant to consist of other matrices as well. In the US, the two dominant models are the 'Matrix of Matrices' and its subset, the 'Clausing Four-Phase Model' (ibid).

Let's now give a brief overview of the HoQ as described by Cohen (1995). The HoQ shows the relationships between the **customer needs** and the **technical responses**. Both the customer needs and the technical responses are drawn up in the form of hierarchies. The impact **relationships** between the customer needs and the technical responses are filled in using

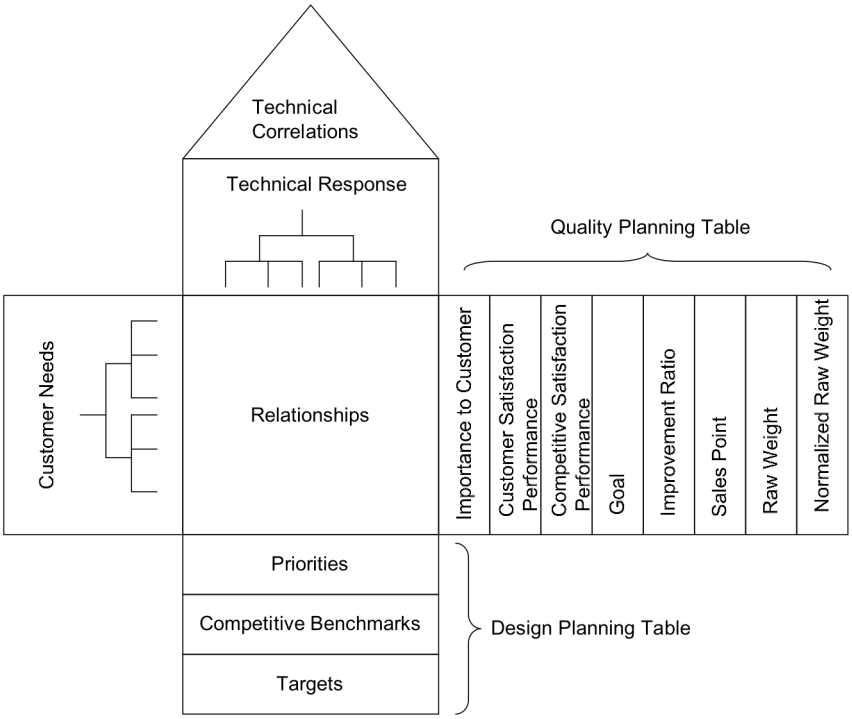

Figure 1. QFD's HoQ. Modified from (Cohen 1995) by showing hierarchical structures for Customer Needs and Technical Response.

impact symbols, where blank = 0 = no linkage, triangle = 1 = weak linkage, circle = 3 = strong linkage, and double circle = 9 = very strong linkage.

Considering the **Quality Planning Table**:

The **importance to customer** values are established on a five-point scale (1-5). These values are found in one of three ways (Cohen 1995):

- By survey: asking the customer to provide a (1-5) value, or equivalent information, which can be translated into a number. This is ordinal scale data, which is not ideal for subsequent arithmetic (see later)
- By using the Analytic Hierarchy Process (AHP): asking the customer to do pair-wise comparisons between all the customer needs. The benefit being ratio scale numbers result that better support subsequent arithmetic. However, the problem is that the number of comparisons grows very quickly as the number of customer needs increases
- By ranking in order: which results again in data on an ordinal scale. If there are numerous customer needs, then the range of the ordinal numbers will be greater than is the case of using survey (1-5) or AHP information, and this has an even greater effect on subsequent arithmetic placing extra emphasis on the more important customer needs (Cohen 1995, pages 99-100).

Likewise a 1-5 ordinal scale is used for **customer satisfaction performance** and **competitive satisfaction performance**. These values are established by survey.

Goal is the level of customer performance that the developers want to achieve, and is set on a 1-5 ordinal scale. It considers the competitive situation and the amount of improvement required.

Sales point considers whether the customer need is likely to help selling the product (1 = No, 1.2 = Medium, 1.5 = Strong).

Raw weight for customer need = importance to customer x (goal / customer satisfaction performance) x sales point.

Considering the **Design Planning Table**:

The relationships are completed by calculating as follows:

> **Relationship** of technical response to customer need = numerical value of impact relationship (described previously) x normalized raw weight for customer need.

The relationships are then added down the technical response column to get a total contribution for the technical response and normalized across all the contributions - the larger the contribution, the more importance of a technical response for the customer. These are called the **priorities**.

The priorities are compared to the competitors' offerings (**competitive benchmarks**) including world-class products, and the development teams performance (which sometimes appears as another box in the design planning table), and then **targets** are derived for the technical responses. The targets are either calculated using mathematical calculation (assuming linear or quadratic relationship) or set using a points system, by taking into account the customer satisfaction performance. The targets are each given a value (using a range of 0-5 – another ordinal scale).

See Figure 2 for a typical published QFD table. There are some differences in terminology and format: 'importance to customer' is called 'priority', 'priority' is replace by 'technical evaluation', the development teams performance appears as 'technical difficulty', and 'target' becomes 'importance rating' and has not been translated to a 0-5 scale. In addition, much of the quality planning table information is not present. Also in the design planning table, 'target values' are added giving numeric quality levels for the design requirements to meet. Such target values are a definite improvement.

Problems with the use of QFD

Let's now explain some of the problems that can be observed in the use of QFD. Problems such as *misuse* of the QFD method will *not* be covered here, given that the main purpose of this paper is to outline how certain IE concepts could be used to improve QFD. However, readers should note that there are many misconceptions about the QFD method and misuse of it (such as comparing the wrong data types in the HoQ) is not uncommon.

Problem: Dealing with customers rather than stakeholders. QFD has a 'customer' concept, rather than dealing with stakeholders. This reflects the origins of QFD, which were to ensure that the customer concerns were translated into the product design. The issue is that in systems engineering, 'customer' is only *one* type of stakeholder. There is a potential risk that users of QFD could inadvertently fail to cater for the many *stakeholders,* who have requirements: 20 to 60 stakeholders are a normal expectation for a large project. 'Missing' stakeholders will have specific requirements that could link to *known* requirements, or could be *new* requirements. The result could be that QFD becomes unable to support us in intelligent prioritization, as it doesn't have the information about *all* the stakeholders, their values and priorities. This is potentially *critical* to project success. The solution is straightforward: replace 'customer' with 'stakeholders' to ensure projects think beyond 'customer'.

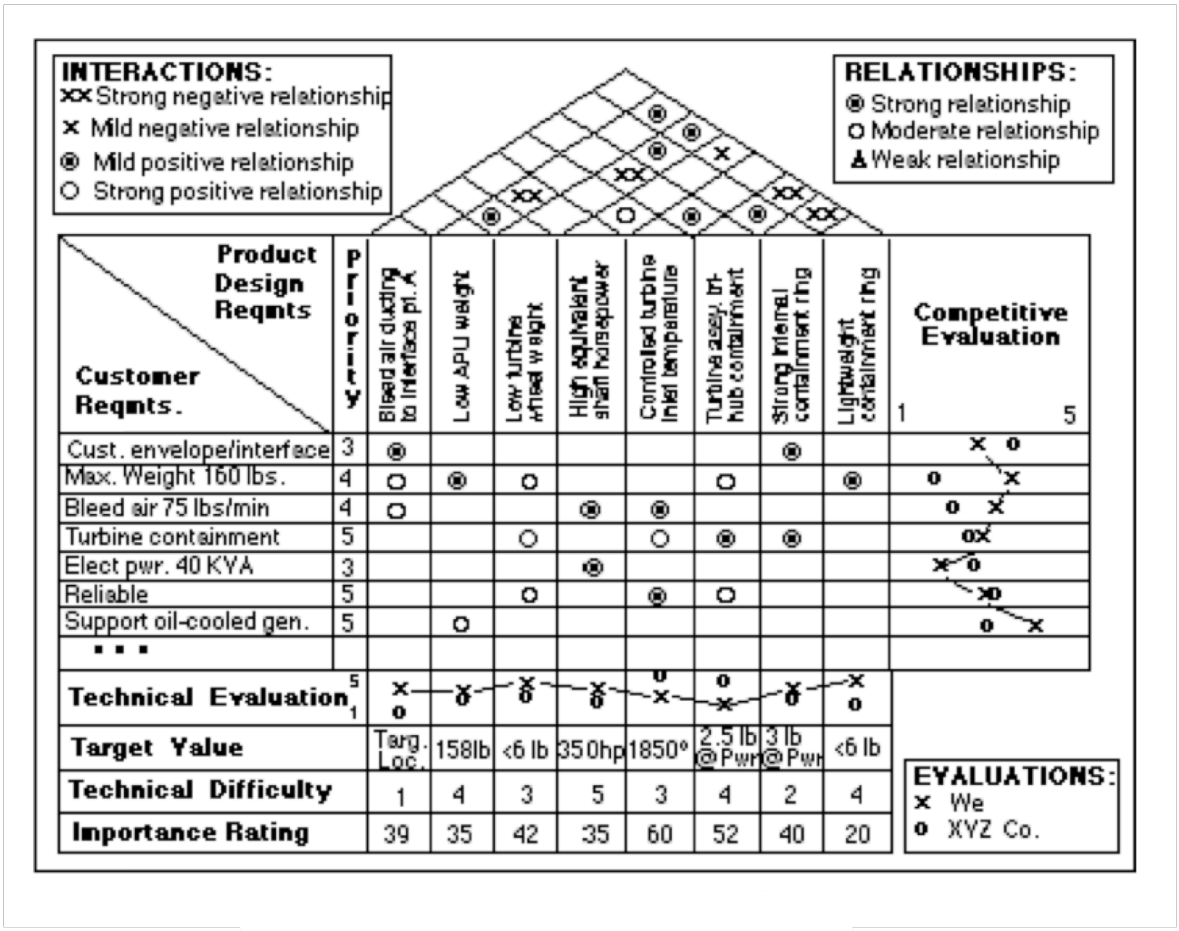

Figure 2. A typical example of a QFD table.

Problem: The requirements are often not sufficiently well defined. Some QFD matrices include target levels (for example, 'Target Value' in Figure 2) and some also include current levels, but most traditional QFD matrices do not contain any numeric levels (Example: 'Reliable'); and rely instead on qualitative requirement statements (for example, see the requirements in Figure 3). One solution would be for QFD to move to consistently using Planguage metrics (explained later). There are other related problems such as not distinguishing between targets and constraints, that we choose to not cover in detail here.

Problem: Use of ordinal and ratio scales. This has already been mentioned briefly in the description of QFD. Carrying out arithmetic on ordinal scales leads to problems, because ordinal scales merely inform you of the ranked order, they do not tell you anything about the size of the difference amongst the items (Cohen 1995). This means any multiplication involving items on an ordinal scale will give questionable results, because it is not known if the ordinal values are strictly proportional.

Looking at the QFD method as described earlier, ordinal data is present as follows:

- Relationships expressed as 0, 1, 3, and 9 (replacing with numbers, the symbols: blank (not linked), triangle (possibly linked), circle (moderately linked), and double circle (strongly linked))
- Importance to customer as 1-5 ratings if gathered by survey or by use of rankings (that is, not using AHP)

- Customer satisfaction performance and competitive satisfaction performance survey information collected as 1-5 ratings.

DDP / Requirements	Hi Rise Escape System QFD	Priority	Activation Time	Communication wi th Central Alarm System	Throughput (X persons/Hr)	Size of lifeboat	Number of steps required to set up	Strenght	Time to Install/Maintain	Operational Independence	Material Cost	Interoperability	MTTR
Cost	Maintenance	2							H				
	Cost of Procurement	3									H		
	Installation	3							H				
Safety	Appearance	3							M	H			
	Accomodation of all body Types	3			L	H							
	Tied to Central Alarm System	3		H									
	High Probability of Servival	3						H					
	Conforms to All Government Regulations	3				M	L					H	
	Ease of Training	1					H						
Versitality	Compatable with Emergency Equipment	2										H	
	Deal with Multiple Incidents	3			M								
Maintenance	Minimal Periodic Maintenance	2											H
	No Interference with routine building maintenance	2							L	H			
Performance	Throughput	3			H			L					
	Ease of Use	1	H				H	L					

Figure 3. Another example of a QFD matrix.

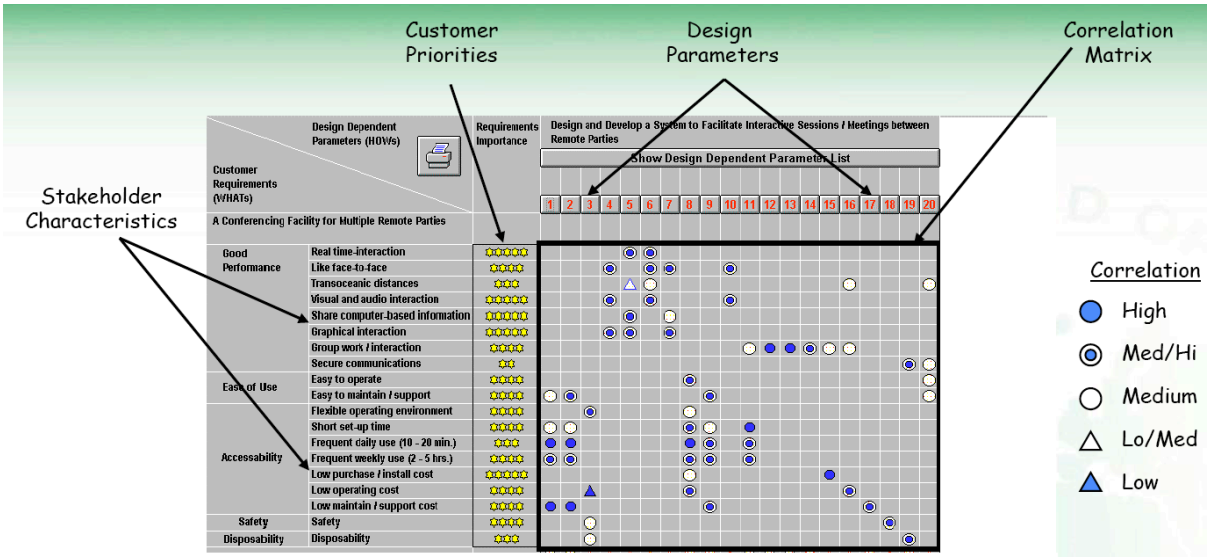

Figure 4. Another example of a QFD Table. The term 'stakeholder' is used, which is better, but 'customer' creeps back into the priorities!

Somewhat better than ordinal scales are ratio scales: ratio scales do tell you the magnitude of the difference amongst items. AHP results in ratio scale data and is sometimes used to determine the 'importance to customer' data. However, even this data must be questioned. If AHP is used, then pair-wise comparisons have to be made. The first question is whether the requirements of AHP for hierarchical structuring of the requirement have been met. Saaty recommends a maximum of 7 items in a group of items for pair-wise comparison (Saaty 1990). In addition, the comparison being made must be a valid comparison to make. Also, are people carrying out the pair-wise comparison interpreting what is being asked in the *same way* and are the *justifications* for the decisions being adequately captured? If there are no *quantified* target levels and/or values supporting the target levels, then such comparisons immediately become somewhat fuzzy. Also people are forced to express their preferences using a 0-9 scale.

Impact Estimation (IE) (Gilb 1998) would argue to use, and retain in the specs, the absolute 'real world' values: then not only is the basis of what information is being considered/compared transparent, but communication of the *real* data is on-going, and thus is reinforced in people's minds. The use of ordinal and ratio scale categories can be *eliminated* (see later explanation of IE priority).

Problem: Not designing with sufficient regard to resources (resource availability, development costs, product price, and operational costs and deployment as well as quality levels): QFD needs to more strongly account for the resources needed to achieve the quality levels, and not focus on quality without regard to people, costs, and product price. Akao (1997) states, "New product development will also require 'price design' along with 'quality design'. Recently Professor Mochimoto has proposed 'QDm', his new concept and method in this attempt. In the future, cost design and deployment method need to be developed, in addition to price design."

Though there is some functionality to handle resources, users of traditional QFD do not *normally* try to deal with resources such as:

- Time to market
- Engineering work hours
- Capital costs
- Life-cycle costs (maintenance and decommissioning)
- Other potentially scarce resources, such as memory space in a product device, such as a mobile phone.

This makes it impossible to *realistically evaluate* and compare the efficiency (impacts/costs) of different proposed design ideas. It also makes it impossible to relate the design costs to the potential stakeholder value we might gain.

Of course, some of these problems might be resolved if people used QFD beyond the HoQ. Additional issues then become the amount of effort people are prepared to invest in carrying out QFD, and the arithmetic problems discussed earlier.

Impact Estimation (IE)

Impact Estimation (IE) was developed by Tom Gilb based on his earlier work with Weighted Ranking by Levels and MECCA (Multi-Element Component Comparison Analysis) (Scharf (aka

Gilb) 1968), and also on Boehm's Requirements/Properties Matrix (Gilb 1976. Boehm's original lecture was at IFIP (International Federation for Information Processing), Stockholm, 1974). IE is one of Gilb's Planguage (Planning Language) methods and uses the language, Planguage, to express its concepts. A main feature of Planguage, in this papers QFD context, is its use of real world quantification. Table 1 shows an example of a simple IE table.

Table 1. Simple IE Table illustrating some of the components of an IE Table. The two proposed design ideas (Idea 1 and Idea 2) are complementary.

Design Ideas-> Requirements: Goals and Budgets	Idea 1 Impact Estimates	Idea 2 Impact Estimates	Sum for Requirement (Sum of Percentage Impacts) (3)	Sum of Percentage Uncertainty Values (4)	Safety Deviation (5)
Reliability 300 <-> 3000 hours MTBF	1950hr (1650hr) ±0 (1)	1140hr (840hr) ±240	92%	±9%	-108%
	61%±0 (2)	31%±9%			
Usability 20 <-> 10 minutes	19min. (1min.) ±4	14min. (6 min.) ±9	70%	±130%	-130%
	10%±40%	60%±90%			
Maintenance 1.1M <-> 100K/year US$	1.1M $/Y (0 K$/Y) ±180K	100K S/Y (1 M$/Y) ±720K	100%	±90%	-100%
	0%± 18%	100%±72%			
Sum of Performance (6)	71%	191%			
Capital 0 <-> 1 million US$	500K (500K) ±200K	100K (100K) ±200K	60%	±40%	-10%
	50%±20	10%±20			
Sum of Costs (7)	50%	10%			
Performance to Cost Ratio (8)	1.42 (71/50)	19.10 (191/10)			

Notes: 1. Time Period: Within next 12 months.
2. Same Safety Margin of factor 2 has been declared for performance requirements and resource requirements. Factor 2 means minimum planned performance requirements > 200% of target (goal), and maximum planned costs < 50% of target (budget).
3. Evidence, Source and Credibility not stated.

IE Requirements. In Planguage, a requirement is defined by the following parameters (See also Figure 5, 'Elementary scalar requirement template') for more options and detail:

- A *tag* giving the requirement a convenient descriptive name or label
- A *scale* of measure giving the units and context for measuring the requirement
- Past levels, under defined conditions (when and where) - benchmarks
- Target levels, under defined conditions (when and where) - Goals
- Constraint levels (worst acceptable levels) under defined conditions (when and where) – worst acceptable cases.
- *Justification* for the suggested targets and constraints. (Voice of Stakeholders)

- Source, authority and status of the information (verifiable power of stakeholders)
- Issues, assumptions, risks, (risk management)
- Value and priority information about the requirement. (Voice of Stakeholders)

Performance (including 'quality') requirements must be defined by 'scales of measure'. A specific system performance requirement can be expressed by more than one scale of measure – sometimes this is needed to fully capture all the aspects of a requirement.

In an IE table, the requirements are placed in the left-hand column. Typically, only the performance and resource requirements (budgets) are used. These are the improvement areas needing design. One aim is to estimate performance-to-cost ratios for the potential design ideas. This is the 'efficiency' (value/cost) of a design. The requirements are usually specified in the IE table by referring to their name tags, and re-stating their primary current and target levels. Other supporting information is available, via the reference tag, in the filled-in requirement specification templates. These can contain any useful quantity of information about a requirement. The level of detail need only be sufficient for current purposes.

In contrast, traditional QFD uses a requirement *name*. However, there is no guarantee, or practice, that this name refers to a well-defined quantified requirement elsewhere. You are left to make your own interpretation of highly ambiguous names like 'Accessibility', and 'Secure Communication'. Such ambiguity is unacceptable in Planguage, and IE tables. More recently Modern QFD (Personal Communication Zultner 2007) has improved on this, and uses a tag, a current level and a target level, which implies a scale of measure. The major difference with IE is that the Modern QFD current and target values are not actually exploited, when deciding the relationships impact data (see later description of IE impact estimates).

Elementary scalar requirement template <with hints>

Tag: <Tag name of the elementary scalar requirement>.
Type: <{Performance Requirement: {Quality Requirement, Resource Saving Requirement, Workload Capacity Requirement},
Resource Requirement: {Financial Requirement, Time Requirement, Headcount Requirement, others}}>.
========== Basic Information ==========
Version: <Date or other version number>.
Status: <{Draft, SQC Exited, Approved, Rejected}>.
Quality Level: <Maximum remaining major defects/page, sample size, date>.
Owner: <Role/e-mail/name of the person responsible for this specification>.
Stakeholders: <Name any stakeholders with an interest in this specification>.
Gist: <Brief description, capturing the essential meaning of the requirement>.
Description: <Optional, full description of the requirement>.
Ambition: <Summarize the ambition level of *only the targets* below. Give the overall real ambition level in 5-20 words>.
========== Scale of Measure ==========
Scale: <Scale of measure for the requirement (States the units of measure for all the targets, constraints and benchmarks) and the scale qualifiers>.
========== Measurement ==========
Meter: <The method to be used to obtain measurements on the defined Scale>.
========== Benchmarks ========== "Past Numeric Values" ====
Past [<when, where, if>]: <Past or current level. State if it is an estimate> <- <Source>.

Record [<when, where, if>]: <State-of-the-art level> <- <Source>.
Trend [<when, where, if>]: <Prediction of rate of change or future state-of-the-art level> <- <Source>.
========== Targets ========== "Future Numeric Values" ===
Goal/Budget [<when, where, if>]: <Planned target level> <- <Source>.
Stretch [<when, where, if>]: <Motivating ambition level> <- <Source>.
Wish [<when, where, if>]: <Dream level (unbudgeted)> <- <Source>.
========== Constraints ========== "Specific Restrictions"====
Fail [<when, where, if>]: <Failure level> <- <Source>.
Survival [<when, where, if>]: <Survival level> <- <Source>.
========== Relationships ==========
Is Part Of: <Refer to the tags of any supra-requirements (complex requirements) that this requirement is part of. A hierarchy of tags (For example, A.B.C) is preferable>.
Is Impacted By: <Refer to the tags of any design ideas that impact this requirement> <- <Source>.
Impacts: <Name any requirements or designs or plans that are impacted significantly by this>.
========== Priority and Risk Management ==========
Rationale: <Justify why this requirement exists**>.**
Value: <Name [stakeholder, time, place, event]: Quantify, or express in words, the value claimed as a result of delivering the requirement>.
Assumptions: <State any assumptions made in connection with this requirement> <- <Source>.
Dependencies: <State anything that achieving the planned requirement level is dependent on> <- <Source>.
Risks: <List or refer to tags of anything that could cause delay or negative impact> <- <Source>.
Priority: <List the tags of any system elements that must be implemented before or after this requirement>.
Issues: <State any known issues>.

Figure 5. A Planguage scalar requirements specification template (Gilb 2005).

IE design ideas. The design ideas are placed across the top row of the IE table by specifying their cross reference tags. Each design should have a corresponding sufficiently-formal design specification, though the level of specification detail can vary, depending on the need, the stage of the design process, and the likely impact and cost of the design. If designs *can* be eliminated early, easily and accurately (for example because of obvious risks, costs, or poor impact) then you don't want to use time obtaining superfluous additional details for them – concentrate instead on enhancing knowledge of the most promising designs. See the design specification template in Figure 6.

Design Specification Template <with Hints>

Tag: <Tag name for the design idea>.
Type: {Design Idea, Design Constraint}.
========== Basic Information ==========
Version: <Date or version number>.
Status: <{Draft, SQC Exited, Approved, Rejected}>.
Quality Level: <Maximum remaining major defects/page, sample size, date>.
Owner: < Role/e-mail/name of person responsible for changes and updates>.
Expert: < Name and contact information for a technical expert, in our organization or otherwise available to us, on this design idea>.
Authority: <Name and contact information for the leading authorities, in our organization or elsewhere, on this technology or strategy. This can include references to papers, books and websites>.
Source: <Source references for the information in this specification. Could include people>.
Gist: <Brief description>.
Description: <Describe the design idea in sufficient detail to support the estimated impacts and costs given below>.

<Term Tag here>: Definition: <Use this to define specific terms used anywhere in the specification>. "Repeat this for as many definitions as you need"
Stakeholders: <Prime stakeholders concerned with this design>.
========== Design Relationships ==========
Reuse of Other Design: <If a currently available component or design is specified, then give its tag or reference code here to indicate that a known component is being reused>.
Reuse of This Design: <If this design is used elsewhere in another system or used several times in this system, then capture the information here>.
Design Constraints: <If this design is a reflection of attempting to adhere to any known design constraints, then that should be noted here with reference one or more of the constraint tags or identities>.
Sub-Designs: <Name tags of any designs, which are subsets of this one, if any>.
========== Impacts Relationships ==========
Impacts [Functions]: <List of functions and subsystems which this design impacts attributes of>.
Impacts [Intended]: <Give a list of the performance requirements that this design idea will positively impact in a major way. The positive impacts are the main justification for the existence of the design idea!>.
Impacts [Side Effects]: <Give a list of the performance requirements that this design idea will impact in a more minor way, good or bad>.
Impacts [Cost]: <Give a list of the budgets that this design idea will impact in a major way>.
Impacts [Other Designs]: <Does this design have any consequences with respect to other designs? Name them at least>.
========== Impact Estimation/Feedback ==========
For each Scalar Requirement in Impacts [Intended] (see above):
Tag: <Tag name of a scalar requirement listed in Impacts [Intended]>.
Scale: <Scale of measure for the scalar requirement>.
Scale Impact: <Give estimated or real impact, when implemented, using the defined Scale. That is, given current baseline numeric value, what numeric value will implementing this design idea achieve or what numeric value has been achieved?>.
Scale Uncertainty: <Give estimated optimistic/pessimistic or real ± error margins>.
Percentage Impact: <Convert Scale Impact to Percentage Impact. That is, what percentage of the way to the planned target, relative to the baseline and the planned target will implementing this design idea achieve or, has been achieved? 100% means meeting the defined Goal/Budget level on time>.
Percentage Uncertainty: <Convert Scale Uncertainty to Percentage Uncertainty ± deviations>.
Evidence: <Give the observed numeric values, dates, places and other relevant information where you have data about previous experience of using this design idea>.
Source: <Give the person or written source of your evidence>.
Credibility: <Credibility 0.0 low to 1.0 high. Rate the credibility of your estimates, based on the evidence and its source>.
========== Priority and Risk Management ==========
Rationale: <Justify why this design idea exists>.
Value: <Name [stakeholder, scalar impacts and other related conditions]: Describe or quantify the knock-on value for stakeholders of the design impacts>.
Assumptions: <Any assumptions that have been made>.
Dependencies: <State any dependencies for this design idea>.
Risks: <Name or refer to tags of any factors, which could threaten your estimated impacts>.
Priority: <List the tag names of any design ideas that must be implemented before or after this design idea>.
Issues: <Unresolved concerns or problems in the specification or the system>.
========== Implementation Control ==========
Supplier: < Name actual supplier or list supplier requirements>
Responsible: <Who in your organization is responsible for managing the supplier relation?>
Contract: <Refer to the contract if any, or the contract template>
Test Plan: <Refer to specific test plan for this design>
Implementation Process: <Name any special needs during implementation>
========== Location of Specification ==========
Location of Master Specification: <Give the intranet web location of this master specification>.

Figure 6. A Planguage design specification template (Gilb 2005). This will give you a fuller

picture of how IE might be described at the level of a single design, before the information is transferred to an IE table, together with other designs.

IE impact estimates. The 'relationship' part of the IE table is filled in by estimating the impacts of the individual design ideas, on the individual requirements. What level, on the defined scale of measure, is this design idea likely to produce when implemented, within timescales and other qualifier (where, if) conditions? The impacts are estimated *quantitatively,* in terms of the estimated resulting level, on the defined scale of measure, for each requirement. This means the impacts are stated using absolute scales of measure.

Each estimated impact level should ideally (or according to corporate standards) be supported by uncertainty estimates (like ±20%) , evidence (facts about impact), source of the evidence, and a resulting (evidence, source) *credibility* level (On a 0.0 (wild guess) to 1.0 (totally credible) scale, how good is the evidence we have for this estimate?) The impacts are then translated into percentage impacts (20% impact), informing us how much each design idea is estimated to contribute towards moving each requirement level from its baseline (current) level (0%) towards its required target level on time and fulfilling any other stated qualified conditions (where, if) (100%). These percentage impacts are (hopefully useful) *rough* estimates, since nothing is specified, known, or considered about:

- any *interactions* amongst the design ideas (the impact could vary if other designs are already implemented, or new designs are added later)
- how difficult it will be to achieve the *remaining* requirement gaps (to 100%) given these designs are implemented.
- the real operational environment, people, organization
- and other factors than can influence the real results.

Notice, that in spite of these limitations on accuracy of initial estimates, we do include as part of the expected Planguage behavior, the notion of evolutionarily collecting real measures of the design impact, even at pilot stages, and feeding that back into the estimates of the impacts ultimately expected, long before full scale commitment to any design is necessary

.

Both the estimated impacts (the absolute real world levels) should be stated in the IE table together with the impact on target percentages, to help you keep a grasp of the situation – the reality, and the degree of completion in relation to targets. See for example 'Usability' in Table 1, the design idea 'Idea 1' reduces time taken down to 19 minutes, which is an impact of moving 10% of the way towards the required target (100%) from the baseline (0%).

The 'percentage impacts', a common currency – despite varied scales of measure, allow some useful arithmetic to be carried out. By summing the columns vertically the 'sum of performance' and the 'sum of costs' can be obtained and a 'performance-to-cost ratio' calculated (see Table 1). The *overall* contribution of the individual design ideas towards meeting all the requirements can be assessed.

These ratios do not tell the 'whole story' as the impact, especially side-effects, on the each one of the individual requirements would also have to be considered (a small impact on one requirement could be offset by a large contribution on another requirement, and this might not be ideal. You might select instead another design idea that had reasonable impacts on all the requirements). This is a tradeoff or prioritization process.

Additionally, the consequent *stakeholder* level value-to-cost ratios ought to be considered (A small improvement in one requirement might be more valuable in *stakeholder* consequence than a large improvement in another requirement impact).

By summing all designs, across the rows, given care is taken to understand whether the sets of design ideas are complementary or alternatives. An idea of whether the individual requirements are likely to be met can be gained. If the design set sum was say 20% then there is probably no hope. At 400% (safety factor 4) we can be more hopeful.

Obviously, these are rough measures, but if the proposed design ideas do not make the required impact, with regard to stipulated safety factor, on various requirements, you know you will probably have to seek additional design ideas. But you can delay this seek until after some realistic evolutionary implementation feedback is gained from the early design implementations, of the most promising impact designs. Maybe additional designs are not necessary after all. Overdesign initially, can drive costs and time to market up unnecessarily.

Let's now consider some examples. Figure 7 gives a performance requirement described using Planguage.

Learning:

Ambition: Make it substantially easier for our users to learn tasks <- Marketing.

Scale: Average time for a defined [User Type: default UK Telesales trainee] to learn a defined [User Task: default Response] using <our product's instructional aids>.

Response: Task: Give correct answer to simple request.

Past [End 2006]: 60 minutes.

GN: Goal [By start of 2009]: 20 minutes.

GA: Goal [By start of 2010]: 10 minutes.

Figure 7. A simple scalar performance requirement expressed using Planguage.

Table 2 shows the information needed for the requirement within an IE table together with additional supporting data, which is not usually displayed in an IE table. The estimated impacts for four potential alternative design ideas are shown.

IE and Risk. IE deals with risk by several means: use of uncertainty and safety margins, and capturing assumptions, dependencies, issues, and known risks.

Evolutionary ('Evo') Plans and Measures. Once the design ideas have been selected, they are sequenced according to the stakeholders' priorities (for specific requirements, including constraints, dependencies, and stakeholder value) and the consideration of actual remaining system developers' resources time, effort, budget) for implementation. An IE table can be used to capture the actual step by step project progress against plan (that is, the *actual* measures after implementation, are put into the IE table, and compared to the estimates. Differences lead to Learning - in a Plan Do Study Act – Deming Cycle)).

Comparison of plans with reality is very important in Planguage. Trying out solutions in an evolutionary manner: that is, implementing high-expected-value solutions at the earliest possible time, and obtaining feedback is key. Estimates need to be tested, and if they are tested early there is still usually time and budget to modify any solutions found to be lacking (even to completely replace them!)

Table 2. An example, which shows the supporting information for an IE table. One requirement is shown with the estimated impacts on it of several alternative potential design ideas.

	On-line Support	**On-line Help**	**Picture Handbook**	**On-line Help + Access Index**
Learning Past: 60minutes <-> Goal: 10minutes				
Scale Impact	5 min.	10 min.	30 min.	8 min.
Scale Uncertainty	±3min.	±5 min.	±10min.	±5 min.
Percentage Impact	110%	100%	60%	104%
Percentage Uncertainty	±6% (3 of 50 minutes)	±10%	±20%?	±10%
Evidence	Project Ajax: 7 minutes	Other Systems	Guess	Other Systems + Guess
Source	Ajax Report, p.6	World Report, p.17	John B	World Report, p.17 + John B
Credibility	0.7	0.8	0.2	0.6
Development Cost	120K	25K	10K	26K
Performance to Cost Ratio	110/120 = 0.92	100/25 = 4.0	60/10 = 6.0	104/26 = 4.0
Credibility-adjusted Performance to Cost Ratio (to 1 decimal place)	0.92*0.7 = 0.6	4.0*0.8 = 3.2	6.0*0.2 = 1.2	4.0*0.6 = 2.4
Notes: Time Period is two years .	Longer timescale to develop			

Planguage and Priority: Planguage has an in-built approach to priority. We never use fixed weights to express priority. We argue these are a bad idea in complex systems (Gilb and Maier 2005).

The Planguage aim is to supply the requirement specification with sufficient information for the priority to evolve, and to be realistically understood according to a reasonable set of real world facts. An IE table has a deadline notion (main project deadline, main product release, unless other deadlines are specified for a particular requirement target or constraint)) when *all* the requirements are meant to be fully satisfied (all the 100% targets met – assuming plans/targets have not been altered). If there are requirements with even *earlier* deadlines then they should have been expressed by using qualifier conditions (for example, [China, End of June 2007]).

Likewise, any *dependencies* for requirements, and all related *functionality* should have been captured. Tradeoffs amongst our set of individual stakeholders have to be made, and that's a question of individual stakeholder value, and total resource availability. Stakeholder preferences ought to map to estimated long term value, under stated assumptions (Expression of stakeholder value is a current research area for one of the authors). This is one of many reasons why Planguage does not ever use weightings, derived from stakeholder preferences (Gilb 2008; Gilb and Maier 2005). There is also a philosophy that if you are rapidly delivering high-value

requirements, then 'rough' is probably 'good enough': there is so much you are going to learn after delivery; and reality is likely to modify your best-laid plans.

Priority is both *complex* (many factors to consider) and *subjective* (who decides which factors to consider, and to what degree at they important?). Common sense dictates that a more realistic priority evaluation process than the QFD/AHP/BSC methods of capturing preferences on 1-5 or 0-5 scales is necessary, in order to reflect real-world dynamically changing priorities, which depend on *many factors*, not just 'customer preferences'.

The priority of a requirement is variable and must be dynamically determined (Gilb and Maier 2005). The *priority* of a requirement is not simply determined by the designs themselves, or even by the impact of the designs on the requirements. The priority of a requirement depends as stated on *many* different factors, and we clearly try to *capture* many of those factors in the scalar requirements template (see above example). Other prioritization factors are attached to the *designs*, and the real world environment (including laws, preferences and competition).

Table 3. A simplified version of part of the IE table shown in Table 4. It only shows the objective, 'Productivity' and the resource, 'Development Cost' for Evo Step 9, 'Recoding' of the Marketing Research (MR) project. On implementation, Evo Step 9 alone initially moved the Productivity level to 27 minutes, or 95% of the way to the target level. Futher weekend work moved it over the Goal level, to 20 minutes. 112.5% og target.

	EVO STEP 9: DESIGN IDEA: 'Recoding'			
	Estimated Scale Level	Estimated % Impact	Actual Scale Level	Actual % Impact
REQUIREMENTS				
Objectives				
Usability.Productivity 65 <-> 25 minutes Past: 65 minutes. Tolerable: 35 minutes. Goal: 25 minutes.	65 – 20 = 45 minutes	50%	65 - 38 = 27 minutes	95%
Resources				
Development Cost 0 <-> 110 days	4 days	3.64%	4 days	3.64%

Table 4. Details of the real IE table, which was simplified in Table 3. The 112.5 % improvement result represents a 20 minutes level achieved after the initial 4-day stint (which landed at 27 minutes, 95%). A few extra hours were used to move from 27 to 20 minutes, rather than use time in the next Evo step.

Current Status	Improvements		Goals			Step 9 Design=>Recoding			
						Estimated impact		Actual impact	
Units	Units	%	Past	Tolerable	Goal	Units	%	Units	%
			Usability.Replaceability (feature count)						
1.00	1.0	50.0	2	1	0				
			Usability.Speed.New Features Impact (%)						
5.00	5.0	100.0	0	15	5				
10.00	10.0	66.7	0	15	5				
0.00	0.0	0.0	0	30	10				
			Usability.Intuitiveness (%)						
0.00	0.0	0.0	0	60	80				
			Usability.Productivity (minutes)						
20.00	45.0	112.5	65	35	25	20.00	50.00	38.00	95.00
			Development resources						
	101.0	91.8	0		110	4.00	3.64	4.00	3.64

Table 5. Real Case from week 9 of 12 before (right side) and after (left side) release to worldwide customers of the product 'Confirmit' from a Norwegian Opinion Survey tool supplier (Gilb & Johansen 2005). The column "Improvements %" shows the result of the last week's design integration, as a percentage of the target level for the release. 112.5% means we delivered 12.5% more than the target level required. The initial 95% progress in 4 days, was worked on a bit more to get to 112.5% by the beginning of cycle 10.

Table 6 Below: the 25 performance requirements actually used in the first 12 weeks of evolution of the product.

Impact Estimation Table: Reportal codename "Hyggen"

Current Status	Improvements		Reportal - E-SAT features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Intuitivness (%)		
75,0	25,0	62,5	50	75	90
			Usability.Consistency.Visual (Elements)		
14,0	14,0	100,0	0	11	14
			Usability.Consistency.Interaction (Components)		
15,0	15,0	107,1	0	11	14
			Usability.Productivity (minutes)		
5,0	75,0	96,2	80	5	2
5,0	45,0	95,7	50	5	1
			Usability.Flexibility.OfflineReport.ExportFormats		
3,0	2,0	66,7	1	3	4
			Usability.Robustness (errors)		
1,0	22,0	95,7	7	1	0
			Usability.Replacability (nr of features)		
4,0	5,0	100,0	8	5	3
			Usability.ResponseTime.ExportReport (minutes		
1,0	12,0	150,0	13	13	5
			Usability.ResponseTime.ViewReport (seconds)		
1,0	14,0	100,0	15	3	1
			Development resources		
203,0			0		191

Current Status	Improvements		Survey Engine .NET		
Units	Units	%	Past	Tolerable	Goal
			Backwards.Compatibility (%)		
83,0	48,0	80,0	40	85	95
0,0	67,0	100,0	67	0	0
			Generate.WI.Time (small/medium/large seconds)		
4,0	59,0	100,0	63	8	4
10,0	397,0	100,0	407	100	10
94,0	2290,0	103,9	2384	500	180
			Testability (%)		
10,0	10,0	13,3	0	100	100
			Usability.Speed (seconds/user rating 1-10)		
774,0	507,0	51,7	1281	600	300
5,0	3,0	60,0	2	5	7
			Runtime.ResourceUsage.Memory		
0,0	0,0	0,0		?	?
			Runtime.ResourceUsage.CPU		
3,0	35,0	97,2	38	3	2
			Runtime.ResourceUsage.MemoryLeak		
0,0	800,0	100,0	800	0	0
			Runtime.Concurrency (number of users)		
1350,0	1100,0	146,7	150	500	1000
			Development resources		
64,0			0		84

Current Status	Improvements		Reportal - MR Features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Replacability (feature count)		
1,0	1,0	50,0	14	13	12
			Usability.Productivity (minutes)		
20,0	45,0	112,5	65	35	25
			Usability.ClientAcceptance (features count)		
4,4	4,4	36,7	0	4	12
			Development resources		
101,0			0		86

Current Status	Improvements		XML Web Services		
Units	Units	%	Past	Tolerable	Goal
			TransferDefinition.Usability.Efficiency		
7,0	9,0	81,8	16	10	5
17,0	8,0	53,3	25	15	10
			TransferDefinition.Usability.Response		
943,0	-186,0	######	170	60	30
			TransferDefinition.Usability.Intuitiveness		
5,0	10,0	95,2	15	7,5	4,5

There are four small teams, each working on one block of requirements. This is the 9th of 12 evolutionary steps before product release to all customers. You can see on "Improvements - %" column how well the teams are doing with 25% of time resources in hand. They are ahead of schedule. The numbers were measured independently by a client (Microsoft, Usability Labs). At this stage we know a lot about the real impact of the designs, and consequently the real need for further designs.

Comparison of QFD and IE

Let's consider how these two methods, QFD and IE compare. The similarities between QFD and IE include:

- Both use a matrix putting requirements against technical response
- Both can cope with complementary and alternative designs
- Both make some attempt to define requirements beyond a name.
- Both make an attempt at initial estimates of the power of designs

The differences between QFD and IE include:

- QFD uses a series of matrices, whereas IE tends to use just one matrix. However, IE can use a hierarchy of matrices for a complex system. For use in the IT industry, QFD would demand the HoQ matrix is done by 'customer needs against functional requirements' (Personal communication Zultner 2007). IE puts requirements against design ideas. As such, IE is very flexible:
  - IE can be used at different levels: strategic planning and tactical planning
  - IE can be holistic in its choice of design ideas, mixing organizational ideas and technical ideas with ease.

  QFD would use different matrices (not HoQ) to tackle design.
- QFD *sometimes* specifies target values for the designs to reach. IE always specifies target values for each of the performance requirements. The corresponding IE impact estimates then express how much the proposed design ideas are likely to achieve towards meeting each of the requirements levels, under specified conditions.
- For risk evaluation, IE captures uncertainty and credibility information to help assess the impact estimates, and uses a safety margin (calculating safety deviation)
- IE supports Evo planning and monitoring. In fact it assumes we can rely on evolutionary feedback to adjust our decisions and priorities.

Summary

It is time to improve the clarity of QFD's articulation of the requirement, design and relationship concepts, so that its analysis is more powerful and realistic. We believe that QFD could benefit from adopting Planguage metrics, and other Planguage constructs.

Maybe Planguage could benefit from some of the rich totality of QFD culture, and we are in the process of looking for useful ideas there now. Unfortunately, as we have learned in this paper preparation, much of the best and updated practices of QFD have been in practice hidden from the broad public view. But the QFD community is a 'learning' community – always looking for improvements.

References

Akao, Y., "QFD: Past, Present, and Future", *Proceedings of the International Symposium on*

QFD '97 – Linköping, 1997.
Clausing, D., "Quality Function Deployment (QFD): Listening to the Voice of the Customer", Massachusetts Institute of Technology, Center for Advanced Engineering Study, 1992.
Gilb, T. and Johansen, T., "From Waterfall to Evolutionary Development (Evo): How we rapidly created faster, more user-friendly, and more productive software products for a competitive multi-national market", *Proceedings of the INCOSE Conference*, 2005. Available from http://www.gilb.com/community/tiki-download_file.php?fileId=32
Gilb, T. and Maier, M., "Managing Priorities: A Key to Systematic Decision-Making", *Proceedings of the INCOSE Conference,* 2005. http://www.gilb.com/community/tiki-download_file.php?fileId=60
Gilb, T., *Competitive Engineering, A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*, Elsevier Butterworth-Heinemann, 2005. ISBN 0750665076, Sample chapters can be found at http://www.Gilb.com as follows:
Chapter 5: Scales of Measure:
http://www.gilb.com/community/tiki-download_file.php?fileId=26
Chapter 10: Evolutionary Project Management:
http://www.gilb.com/community/tiki-download_file.php?fileId=77
Gilb, T., *Software Metrics*, Studentlitteratur (Sweden), 1976. Also Winthrop (USA), 1977.
Gilb, T., "Impact Estimation Tables: Understanding Technology Quantitatively", *Crosstalk*, December 1998. http://www.stsc.hill.af.mil/CrossTalk/frames.asp?uri=1998/12/gilb.asp
Gilb, T., "Choice and Priority Using Planguage: A wide variety of specification devices and analytical tools", Submitted to INCOSE 2008, http://www.gilb.com/community/tiki-download_file.php?fileId=48
Saaty, T. L., "How to Make a Decision: The Analytic Hierarchy Process", *European Journal of Operational Research,* 1990.
Scharf (Gilb), T., "Weighted Ranking by Levels", *Norddata Proceedings* (Copenhagen, Helsinki), 1968-1969. Also in the *IAG Journal* (IFIP Administrative Group, Holland), 1969 and in Gilb, T., *Controlling the Computer*, Studentlitteratur AB, Lund, Sweden, 1974.

Biographies

Tom Gilb is an international consultant, teacher and author. 'Competitive Engineering' is his ninth book. Tom has worked with major multinationals such as Credit Suisse, Schlumberger, Bosch, Qualcomm, HP, IBM, Nokia, Ericsson, Motorola, US DOD, UK MOD, Symbian, Philips, Intel, Citigroup, United Health, Boeing, Microsoft and Telenor.

Tom's website (http://www.Gilb.com) has a large number of papers, slides, book manuscripts, case studies and other artifacts, which would help the reader learn more about Planguage.

Lindsey Brodie is currently studying for a PhD. at Middlesex University, UK. She holds a MSc. in Information Systems Design. She edited Tom Gilb's book, 'Competitive Engineering'. Previously she worked for many years for International Computers Limited (ICL), UK carrying out technical project support, product support (operating system and database support), and business process and management consultancy.

Lindsey is a member of the British Computer Society and a Chartered Engineer (MBCS CITP CEng). Email: L.Brodie@mdx.ac.uk

Version Nov 22 2007: 15:48-18:38 tg minor edits vs 1833 version, tracking on but not visible.