Developing World Class Product Requirements
And how Six Sigma tools can help

Samuel J Keene
Fellow of the Institute of Electrical and Electronics Engineers
s.keene@ieee.or

Specifying product requirements seems so simple. Don't we all know what we want? Isn't the customer always right? Apparently not, at least not at first. Requirements are the foundation of product development. So when the requirements are flawed, all the downstream processes are likely to be flawed also. Typically, requirements deficiencies do not seem to get caught until the product reaches the end customer. That is the worst time to catch faults and the most expensive to the developer. The cost to correct a defect increases exponentially over program phase or time. So the most expensive system defects are the requirements problems that are found in the field after delivery. They are also the most difficult faults to fix. The defects have been in the design from the beginning and later design aspects have been dove tailed to the initial requirements, which have now been found to be lacking. The cost of changes during the requirements phase is trivial. The cost of changes in the field can be so high as to destroy a company.

Well, what is so difficult about specifying product requirements? There turn out to be a number of entrapments. We will discuss some of these challenges and then introduce tools that can help identify, mitigate, or prevent these potential problems. Requirements problems arise from:

- Misunderstanding needs
- Incompleteness
- Ambiguities
- Interpretation vagaries
- Conflicting goals
- Requirements creep
- Natural language vagueness and interpretations
- Not satisfactorily dealing with variations in the environment
- No buy-in from stakeholders
- Too volumness; not focused
- Lack of customer awareness and reality
- Not accounting for "lessons learned" on past programs
- Lack of situational awareness (interfaces: hardware, user, applications, environment)
- Not accounting for regulatory compliance requirements

The requirements improvement discussed in this section are drawn from the Six Sigma toolset. These are tools that everyone can readily use. Statistically, Six Sigma represents 4.3 failures per million opportunities. The Six Sigma quality level really represents a world class process level, or the best of breed. Obviously, Six Sigma is a statistical concept, but it is much more than that. The tools that we will use in this section are all descriptive or analytical; they are not statistical. They have all existed before Six Sigma program was formalized around 1990, but now have been adopted by Six Sigma. These tools all aid in discovering and refining product requirements. Good requirements must consider and aid in satisfactorily managing contingent conditions that arise. The straight line code operation always works perfectly. It has been exhaustively tested. Good code must go further and satisfactorily manage all that aberrant conditions that arise in the program or the machine environment.

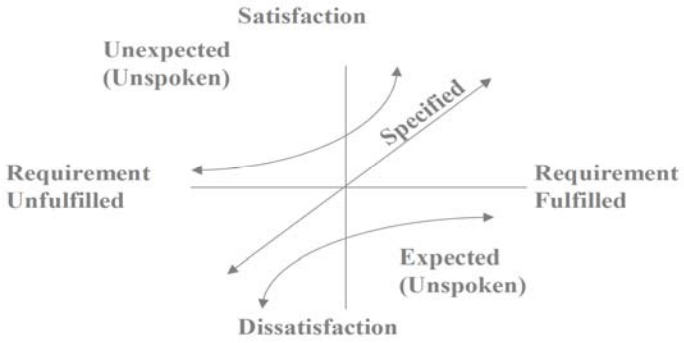
Most often the customer releases product requirements before they have been fully thought out. Product requirements still have to be developed sufficiently to satisfactorily handle the breadth of the application or environmental variations to which the product is subjected. The initial requirements jump right out. It is the tip of the iceberg that always comes easiest. Like Paul Harvey says, the “rest of the story” takes digging to fill out the requirements. Like the axiom goes, “the devil is in the details”. That's where the hard work must take place. The business analyst or the Six Sigma black belt can meaningfully work with the designer to help fill out the requirements. This collaboration process can be supported by Six Sigma collaborative design tools. The analyst and the designer can team up to explore new views (via Six Sigma tools) of the system. The analysis exercises shown in this section are performed by the analyst working with the designer. In every case, that I have been involved with, the designer has either made a change to his design or wished that he had thought of this nuance earlier. This type of analysis always leads to a design improvement or at least provides a validation of the design. Either situation is valuable.

Note that the collaborative tools we use are not necessarily new to Six Sigma. Many have been around for some time and furiously adopted by Six Sigma. These include Potential Problem Analysis [1], FMEA[2], and Goal-Question-Metric (GQM)[3] to name the first three that I would recommend. The first two are old friends to assurance engineers. Certainly PPA and FMEA precede the formalization of Six Sigma. GQM was developed in the Six Sigma timeframe. All three of these tools will be discussed in this section. Let’s start at the beginning. Requirements articulation traditionally comes from the expressed customer “needs”. These traditional requirements are what we think of when we think of “customer requirements”.

The concept of “context data” to supplement and augment “needs” requirements has big payoff potential. “Context” data is contrasted against the “needs” data, where the latter is the articulated needs of the customer. Contrastingly, context data is more like a storyboard with the contributors being the product's stakeholders. For example, I asked a successful realtor the reason for her success. She told me that her success was based on hearing “what the customer did not say” (at first). She was looking for the underlying story that was driving the stated requirements. She engaged the customer in a conversation and listened for the customer’s underlying preferences. The product preferences and concerns that she was particularly looking for are those that the customer had not conceptualized yet or mentioned to her. This is referred to as “context data” about customer needs and wants. Context data can also come from developing a deeper understanding of the customer basis for his needs data. It can also come from other product stakeholders, who have different views and experiences with the predecessor products. The deeper understanding may enable that the developer to find a more satisfying requirements solution for the customer and also avoid some potential dissatisfiers the customer may have with the product. This is a good time to introduce Kano requirements model [4]. See Figure 1.

Kano Customer Satisfaction Model

Figure 1: Customer Fulfillment: Kano Diagram



The Kano model is a theory of product development and customer satisfaction developed in the 1980s by Professor Noriaki Kano which classifies customer preferences into three main categories:

- *One-dimensional Quality (specified needs).*This is based upon “needs” data. These attributes result in satisfaction when fulfilled and dissatisfaction when not fulfilled. The customer’s satisfaction is directly proportional to the degree of product fulfillment, as perceived by the customer. The more the customer gets, the more pleased he is. An example would be the lifetime of a light bulb. Ordinarily light bulbs typically last a lifetime of 700 hours. If they demonstrated only 300 hour life time, the customer would be dissatisfied. Conversely, a bulb that demonstrated 2000 hour lifetime would please the customer proportionately. This is represented on the Kano diagram by the straight line that runs at 45°. The longer the observed life time of the bulb, the more pleased the customer will be. I bought a case of light bulbs once. I think there were 96 in the case. The advertised life time was 1,000 hours. They burned out much faster than that.

- *Must-be Quality (potential dissatisfiers if not met).* These attributes are often taken for granted when fulfilled but result in dissatisfaction when not fulfilled. This Kano model characteristic is the “expected but unspoken” customer needs. These are qualities that the customer assumes are present, even though they were not stated, and the customer is very dissatisfied when these product capabilities are lacking. A personal example of this was a new car I bought and later discovered the car was lacking in braking capacity. It took too long to stop and was only discovered in an emergency braking situation. The worst time to find this out. This capability was never considered or included in the decision set criteria for buying a car. The “must be” quality characteristic is shown on the bottom half of the Kano graph (3rd and 4th quadrant). It is basically a dissatisfaction curve. When the customer finds this product characteristic lacking, the customer is negatively impacted. It is not even necessarily noticed or appreciated when it's present. The risk is only on the downside.

Infrastructure trustworthiness should be a major concern for the product producer. There are a lot of assumptions about the infrastructure mainly that it will just be there and is assumed to be available and trustworthy; often it is not even thought of. It is just expected. Infrastructure is most often only noticed in its absence. A well designed system will identify and validate these infrastructure assumptions. There also needs to be contingency plans to manage the system when there are abnormal input parameters; faults arise in the infrastructure, or environmental threats. Broken functions, faulty infrastructure, or untrustworthy performance can lead to great customer dissatisfaction. The author had a car with a faulty air conditioner. I took it into the dealership for repair. They had to order the part which took a month. This long lead time was common whenever any custom part had to be ordered. When I brought the car in a month later it took two days to make the repair. When I came to pick up the car, I stood in line at the cashier. The cashier told me when I got to the front I had to go in the shop to get the bill. Entering the shop, I found the mechanics playing frisbee on the shop floor. So I had to walk around that distraction. Then I finally retrieved my bill and went back to the cashier to pay it. Then when I turned on my car, smoke was billowing out from the engine. I drove that home because I did not feel I had any other choice at that time. People were honking at me, thinking I was on fire. I took it back and got it repaired properly. I later wrote a note expressing my dissatisfaction with this repair process. Six Sigma analysts are very sensitive to four processes. This was one. Years later I considered buying a car of the same manufacturer type. My wife would not let me do that. She remembered how bad the service experiences been. This was the continuing manufacturer impact from that dissatisfying experience. It was more precisely a bad dealer experience, but we generalized to discredit the car manufacturer.

- *Attractive Quality (Customer delighters).* These attributes “delights” the customer when fully achieved, but is not noticed or expected when it is not fulfilled. These are attributes that are not normally even thought of by the customer. The customer is surprised and delighted when he discovers this unspecified attribute quality. The attractive quality is shown in the upper half of the graph. That is quadrant to 1 and 2 of the Kano diagram. This attribute can only be a winner. There is no downside. These product

delighters can often be positive differentiators to the customer's eyes. They can sell the customer. It is exciting to find these underlying customer delighters. Three examples follow:

1. I had lunch the Vice Pres. for General Motors Quality. During which, he mentioned that he would often test ride competitors cars to pro actively compare to their own cars. He was looking for the delighters that the competitors have put into their cars. It takes effort to find the delighters. One time he was sitting in the backseat of a competitor's automobile. Then, when he prepared to get out of the backseat he reached for the top of the door and he found a handle above the door. He did not expect this. It was a pleaser. We can sometimes identify underlying exciters by engaging the customer in conversation like the realtor did. All the time looking for insights on what mattered to the customer that he had not told her. We might be able to improve the requirements statement as we better understand the underlying customer needs. The added perspective might hone the requirement specification making it truly reflect the customer need.

2. A proactive executive would take his staff to visit competitor's stores. They were looking for things that their competitors did better than they did. The executive was not interested in any of the faults of the competitor store. It was as if he did not see them. Any faults in the store were of no concern to the executive. He took no pleasure in his competitor's deficiencies. He was only seeking to make his business better.

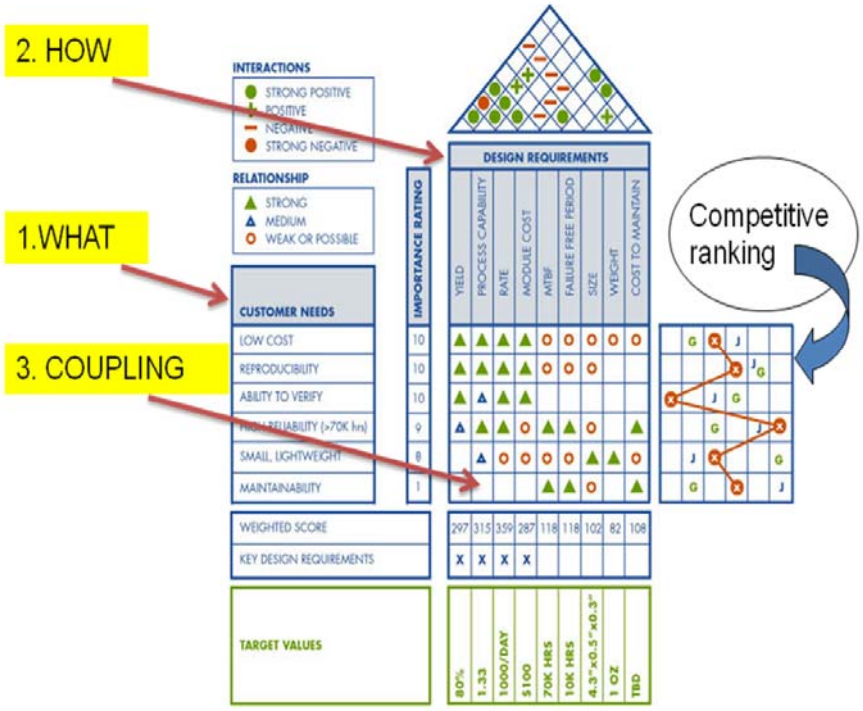
3. There is a creative business development process called "customer centered quality". This was sent to me in a private article [5]. This process can help discover and promote customer delighters. Let me describe the example that was included in the referenced article. The McKesson Company used customer centered quality to successfully differentiate itself from others in a commodity market. A McKesson representative would go to a pharmacy and provide the pharmacist with a barcode scanner to aid in taking weekly inventory. The inventory would automatically be sent back to McKesson for stocking. McKesson would then determine an optimal order quantity for the items needed. The drugs shipped to the pharmacist would be boxed in the order of the layout of the pharmacy's shelves. The box contents were tailored to the location of the product on the bookstore shelves. So the pharmacist was able to stock his shelves in a linear fashion. Further, McKesson helped the pharmacist with a database of his customers' prescriptions. Each customer's prescriptions were analyzed for any potential adverse drug reaction. This proactive effort on the part of the pharmacy led to a reduction in the pharmacies liability insurance premium. So McKesson was differentiating itself with respect to the competition by tailoring their supplier interface to be meet the customer needs.

Communications

Communications are always a challenge to make sure what is desired is properly communicated and understood. Communications must flow with fidelity between the stakeholders: end customer, product planners, implementers, testers, marketers, and anyone else who has a vested interest in the final product. Most importantly, there has to be a common understanding between the product planers, stakeholders, and implementers of the design. It has been said, "If you want to make sure someone understands something, ask them to repeat it back to you". This is not sufficient. This parroting back the spoken requirements does not assure understanding and agreement between the stakeholders. It only demonstrates that your words were heard. It is better to inform the implementers "what is wanted" and then let them respond by telling you "how they plan to meet your specified needs this communication process is akin to the "house of quality" requirements management. This process is depicted in the (HOQ) graph in Figure 2. The HOQ is the start point for the Quality Function Deployment (QFD) [6],

We are going to just look at the three fundamental components of the HOQ graph in Figure 2. The HOQ itemizes and shows what the customer "wants" (shown in block "1). The customer needs make up the "What" in the HOW Matrix graph below.

Figure 2: Customer Fulfillment: Kano Diagram



Source:American Society for Quality. ASQ Six sigma Black Belt online learning course, Relase 2.3. ASQ 2006 (built from)

The developer's response to the customer's stated needs is shown in block 2. This itemizes and details the components of the developer's solution to satisfy customer needs. There should be good correlation between the customer needs and the developer's solution. The "House of Quality" matrix shows the correlation between the customer's needs ("wants") being satisfied by the developer's solution ("how's").

The HOQ lists the customer's needs on the left vertical axis. It is possible that one of the customer needs is not being met by developer's solution. That would be illustrated by one of the rows having no coupling with the developer's solution. It would just be an empty row in the matrix. Also you could have an anti-column in the matrix. That would indicate that a component of the developer's solution did not bear on a customer need. Using this "what is needed" met by "how it will be satisfied' approach can be beneficially applied everyday in our requirements' interactions. Try this out with your spouse, children and friends, so you don't have to say later, "Now I see what you wanted". The most severe breakdown in system requirements understanding, that I have seen, could have been averted by just consistently using this HOQ mapping practice.

System Management Challenge

Requirements deficiencies are the main driver of System Reliability Problems, cost over runs, and schedule slips. Brendan Murphy of Microsoft has reported that most field problems on large systems result from design deficiencies in:

1. System requirements, or in
2. Managing system interfaces
3. Managing design changes – Sam Keene addition

Murphy has labeled 1 and 2 above as System Management Problems [7]. This was based upon his tracking over 2,000 computer systems deployed in Europe. This author added number 3 above, reflecting the unintended consequences of making changes to the system. Changes are fraught with risk. When it comes to changes, there is no such thing as a "small" or inconsequential change. The system maintainer often lacks some of the "situational awareness" needed to avoid problems when instituting design and code changes. Often so called "small changes" do not get adequately reviewed and tested. Their consequences can be large. A prime example Software changes degrade the architecture and increase code complexity

Developing better requirements

1. Every requirement must be measurable or able to be validated. It is not adequate to state that the product must be reliable; it must make a quantitative statement, i.e. The MTBF must exceed 200,000 hours.

2. The best requirements are concisely stated. Mark Twain once said, "I would have written you a shorter letter but I did not have the time?" So it is worth the effort to make sure the requirements statements are focused and concise as possible while preserving the requirements content. Pictures, charts, and graphics can effectively convey a 1,000 words and enhance the understanding.

3. Once the customer sees (uses) the product, the customer will be able to better state the likes and dislikes and improvement preferences. It is easier to iterate a design than to start with a new concept and deliver a finished product in one development cycle. This iterative process provides the customer an opportunity to participate interactively with the developer throughout the product development, which also increases the customer buy-in to the final product. Product changes are best accommodated in enhancement release cycles are included in scheduled change cycles. So changes can be planned in, rather than forcing them in at the last moment. Change is always risky but better timing and process management of changes can mitigate the risk. Changes potentially increase design complexity and degrade the architecture. They need to be carefully managed.

4. The developers are often pushed to hurry to get the product developed despite their fuzzy understanding of the requirements. The developer thus proceeds with partial articulation of needs, and even that articulation is flawed in natural language vagaries. The developer also has a gap in domain knowledge vis-à-vis the customer application. So there are a lot of hurdles to overcome. Requirements development is best accomplished as an incremental process. Barry Boehm's "spiral development model" [8] is an incremental (and recursive) adaptation of the classic "waterfall development model" [9]. The spiral model plans to incorporate evolving requirements in a systematic manner.

5. Prototypes can be used to give the customer "hands on" feel of the new product as it is developing. This allows the customer to make timely inputs into the product design team and prepare the users for the new product use. These design iterations follow the so-called "Spiral Development" model, coined by Professor Barry Boehm. (ibid 8). The design requirements are not complete until the things we do not want the system to do are specified, and precluded, in addition to all the things we wish the system to do. For instance, if we are dealing with a heat-seeking missile, we do not want it to seek our

own heat. The best way to find and document the things we do not want the design to do is to do a Potential Problem Analysis or a Failure Modes and Effects Analysis (FMEA) on the design.

6. This author has found it essential to work directly with the designer in analyzing their design. The analyst systematically goes through the design with the designer to document the reaction of likely failure modes posed by the analyst. When done, the FMEA stands as an artifact to the due diligence of the designer to identify and mitigate the likely failure effects. Every time I have performed an FMEA with the designer, the designer has made at least design change from our analysis. Alternatively, the designer wished we had done this analysis earlier when the design could accommodate such a change. The high level FMEA working with the designer typically takes usually a half-day and has been a high payoff activity. The designer has been driven to get something working; this FMEA exercise completes increase the design robustness. The FMEA analyst is a catalyst in helping the designer to take a fuller view of the design. The designer will make changes as their benefit becomes evident during the analysis.

7. Product changes are high risk activities. Changes to released products can best be incorporated when done on a planned A-B cycle. That is, changes are planned for in incremental cycles. Release cycles are planned in advance, say every 3 months (A release followed 3 months later by a B release and so forth) so new changes can be scheduled in and adequately tested prior to release. No last minute changes are allowed if they are not adequately tested.

Mikel Harry, Six Sigma pioneer, said, "Six Sigma is a process of asking questions that lead to tangible, quantifiable answers that ultimately produce profitable results. Samuel Keene further suggests, question:

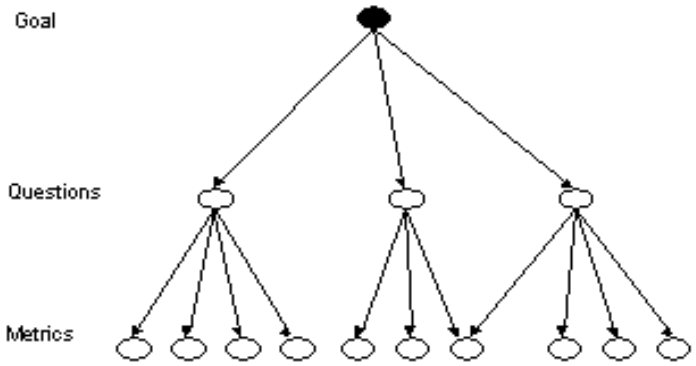
- Data – for its applicability and validity, by demonstrating the measurement systems are repeatable and reproducible, i.e., that they are gage capable. The data should also be representative of the population under study, be of a size that is statistically significant and randomly drawn from the population, to make proper inferences from the sample.
- Process – for its applicability to the design problem and whether it is optimized to the tasks at hand.
- Assumptions – for their currency and applicability and completeness.
- Standards – which the design must meet and comply. Additionally there are requirements to meet business and regulatory compliance needs.

The Six Sigma collaboration facilitates team communication, understanding and discovery. This helps both the customer and the developer identify and quantify product requirements. This process also captures the team thinking and decision making that went into developing the requirements. This will stand as a monument to the requirements development process. This provides traceability of design decisions. One valuable collaborative tool that can beneficially be applied at the project onset is the Goal – Question – Metric (GQM) tool. This analysis tool was developed by Victor Basili of the University of Maryland, College Park and the Software Engineering Laboratory at the NASA Goddard Space Flight Center]. GQM is implemented in three steps:

1. Enumerate the goals for the project. These can be prioritized, associated or nested as best fits the program.
2. Brainstorm questions around each of the goals, or at least around each of the major goals, to help completely define/refine those goals in a quantifiable way.
3. Specify the measures or actions needed to answer those questions and track process and product conformance to the goals.

This GQM flow down process is shown in Figure 3.

Figure 3: Goal – Question - Metric



It can be noted that metrics drive behavior: good metrics drive the right behavior. Good metrics should answer questions of interest. Dr Keene uses the questions as a basis to map appropriate tools to answer those questions, leading to an action plan or project plan.

These measures improve the development process by clarifying product goals helping both productivity and quality. This is a good introspective process tool. It is an enjoyable experience for a team to be engaged in such collaboration. It builds better cross team understanding and cooperation, and more importantly, it leads to discovering more product questions that need to be answered. The collaborative tools should typically focus on those requirements that are new, unique or complex. This is where the greatest opportunity lies. We will use these questions to:

1. Clarify meaning of the goal. This will sharpen what product behaviors are desired vs those that are of lower priority. Sometimes stating what you do not want to the product to do is as important as stating what you want the product to do. This may beneficially lead to a restatement of the goal (requirement). Mark Twain once said, “The difference between the right word and the almost right word is the difference between lighting and lighting bug.”
2. Identify those questions whose answers will strengthen the stated requirement and validate its intent.
3. GQM provides an unintimidating basis to ask questions. We are questioning to learn. Both the person asking the question and the person answering the question stand to benefit. The more questions that students raised in my Six Sigma classes, the cleverer I deem the class or the individual asking them. There are no dumb questions and GQM provides the ready framework to question the basic requirements. All questions have their own clarifying value plus transitional value to trigger the next question to be raised. This is a rich idea discovery setting of “lateral thinking” as promoted by Edward deBono[10].
4. Identify those metrics or experiments or tests to clarify and answer the questions raised. A good metric is anything that answers a question of interest. This step typically amounts to an action plan for particular person(s) to study or analyze or test to provide the answer. A good outcome is to add conditions or increase the accuracy and definiteness of our goal.

Metrics can be made more meaningful it they are tied to questions you are interested in answering. Then better data can be collected if the people are trained and understand the purpose of the data they help to collect. The value of the metrics collection will be enhanced by feeding back to the team the results of their data collection along with the insights gained from analyzing the data.

We need to focus and develop good requirements statements. This does take effort and this can be meaningfully improved by using the Six Sigma collaborative tool set. It is an enjoyable and mind-

expanding experience. These tools promote fuller understanding of the customer's needs, achieve customer buy-in to the requirements development, and leave traceability to the design decisions that are made.

References

1. http://www.diegm.uniud.it/create/Handbook/techniques/List/PPA.php
2. [http://en.wikipedia.org/wiki/Failure_mode_and_effects_analysis]
3. http://www.goldpractices.com/practices/gqm/index.php]
4. http://en.wikipedia.org/wiki/Kano_model
5. http://en.wikipedia.org/wiki/House_of_Quality
6. Brendan Murphy, Ted Gent; Measuring System and Software Reliability Using an Automated Data Collection Process; Quality and Reliability Engineering International Volume 11, 341-353, 1995
7. http://leansoftwareengineering.com/2008/05/05/boehms-spiral-revisited/
8. http://en.wikipedia.org/wiki/Waterfall_model
9. G. Markovits & Associates, Inc., "Customer-Centered Quality – A Key to Competitive Advantage, Wappinger Falls, NY, June 1,1993 (private correspondence)
10. http://www.edwdebono.com/debono/lateral.htm