01 Dec 2000

Software Quality Function Deployment

Xiaoqing Frank Liu

Missouri University of Science and Technology, fliu@mst.edu

Software quality function deployment

Xiaoqing Frank Liu

© Art Today/Jeff Nishinaka Photo Composite: Debbie Cantillo/Jim Hankard

Any product development involves projecting its potential success in achieving its functional and commercial goals. Better quality designs that match customer needs and preferences and integrate other lifecycle issues early in the software development process are more likely to be competitive. Thus, there is significant concern in industry about quality product design, which is addressed by Quality Function Deployment (QFD).

QFD uses matrixes to organize and relate pieces of data to each other. These matrixes are often combined to form a basic tool of QFD, called a House of Quality (HoQ).

QFD was developed in the Kobe shipyards as a way to expand and implement the view of quality as taught by W. Edwards Deming and others. It has been widely applied in many industries worldwide, such as automobile, electronics, food processing, computer hardware and software ever since. Software Quality Function Deployment (SQFD) focuses on improving the quality of both the software development process and the product.

The improvement in software quality leads to fewer changes in requirements specification, design, and code, a reduction in the number of defects and less rework, and therefore, higher productivity. SQFD was first developed to improve the quality of embedded software in Japan. It has been applied to improve software quality in many large organizations, such as DEC, AT&T, Hewlett-Packard, IBM and Texas Instruments.

SQFD has been utilized in developing various types of software products, such as operating systems, embedded software, management information systems, decision support systems, network and transaction processing systems. SQFD has been beneficial in developing new software products and upgrading or enhancing existing software products. It helps to enhance communication between customers and software developers and testers. Therefore, SQFD can be used to improve customer satisfaction.

House of Quality

There are usually six parts in an HoQ, as shown in Fig. 1. Customer's requirements are listed on its left side. The house of quality documents customer requirements from multiple perspectives. It outlines what customers want.

In the customer requirement acquisition, we need to make sure that complete, consistent, nonambiguous, nonredundant, and true customer requirements are identified and specified. There are three types of requirements: revealed requirements, expected requirements and exciting requirements based on Kano's model.

Revealed requirements are typically what we get by asking customers what they want. These requirements are satisfied in proportion to their presence in the product or service. Expected requirements are often so basic that customers may fail to mention them until products fail to perform them. They are expectations, without which the product or service may cease to be of value; their absence is very dissatisfying. Exciting requirements are difficult to discover. They are beyond the customer's expectation. Their absence does not dissatisfy; their presence excites.

Part 2 lists what your company can do technically to satisfy customer requirements. Typically, it lists engineering characteristics of the product, which serve as a basis of engineering design. They should be measurable. Part 3 specifies trade-off relationships between engineering characteristics. There are three types of relationships between them: positively related, negatively related or irrelevant. Two engineering characteristics are negatively related with each other if an increase in the value of one engineering characteristics usually decreases the value of another. Part 4 correlates what customers want from a product and how the company can meet those requirements. It is a core matrix of quality function deployment.

In part 5, the importance rating, and competitive analysis are specified in terms of customer requirement satisfaction, as well as planned level and sales points. The improved ratio is computed based on current and planned levels as follows:

improvement-ratio = planned-level/current-level.

The importance weight of a particular customer requirement is computed as

follows:
*Importance-weight = degree-of -importance * improvement-ratio * sales-point.*
A normalized weight of a customer requirement indicates its relative importance among all customer requirements.

Technical feasibility is assessed and technical competitive analysis is conducted in part 6. The importance weight of an engineering characteristic is computed by adding up its impact importance weights on all customer requirements. An impact importance weight of an engineering characteristic on a customer requirement is computed by multiplying the value of the impact relationship with the normalized weight of the customer requirement.

The house of quality can be used to incorporate customer's voice into every manufacturing activity, as shown Fig. 2. It helps to trace what design engineers and manufacturers do for what customers want.

A framework

Current software manufacturers usually try to capture market share with rapid software development and marketing. However, they usually suffer from poor quality of their software products. Actually, the development of high-quality software products can decrease cost by reducing rework and increasing productivity. Therefore, it enables software manufacturers to capture market share with better quality and lower price.

Software quality can be viewed as conformance to software requirements from customers. Therefore, every software development activity, such as software architecture design, data structure design, coding and testing, should be driven by customer's requirements. However, existing requirement analysis and specification methodologies, such as structured analysis and design and object modeling techniques, have difficulty in clearly identifying how what software designers do impacts on what customers want. In addition, they are awkward in detecting conflicts between customer requirements.

On the other hand, the house of quality incorporates satisfaction of customer requirements into every software development activity, including software architecture design, data structure design, coding, testing and so on. It has been applied in the development of many software products.

A framework for applying the house of quality in software development is shown in Fig. 3. Analysts must make sure that technical features of the system conform to the customer requirements. Technical features that have nothing to do with customer's requirements should be removed from the system specification.

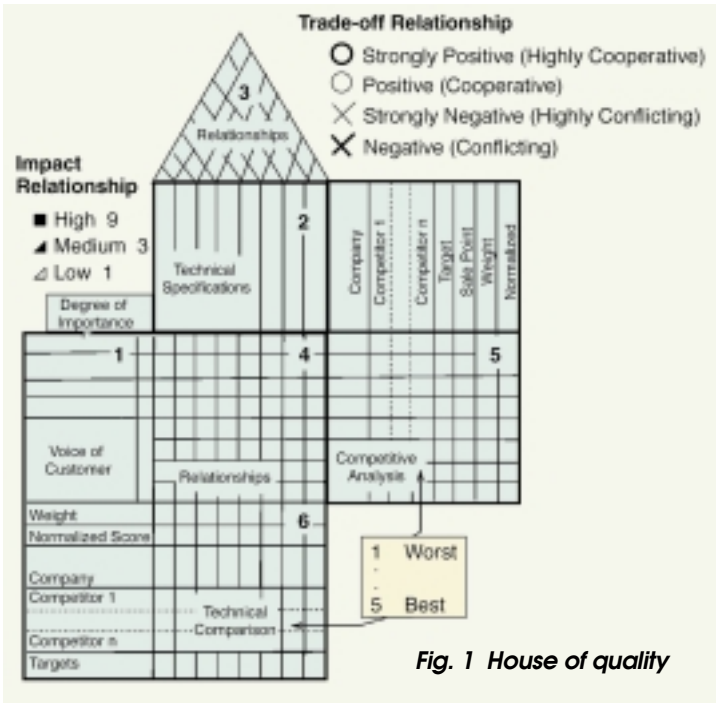

Fig. 1 House of quality

In software design, engineers develop software architecture, a module structure, data structures and user interface according to functional specification and non-functional constraints. Design issues that have nothing to do with the technical features of the system must be removed. In the implementation phase, programming languages and tools are chosen. Programs are developed according to design specification. In the testing phase, test plans are developed and testing is done to remove defects in the programs.

An application example

We present an application example, which is summarized from graduate students' exercises in the software quality course in the University of Missouri-Rolla. Supposed that we need to evolve a distributed information system for a hypothetical company. First, let us solicit customer requirements. Five customer requirements in critical areas of quality, cost and schedule are identified. They include:

- it should respond quickly;
- it should be reliable;
- it should be accessed from multiple sites;
- it should be developed on schedule, and
- it should be developed within the budget.

These requirements are specified in the left side of a house of quality (Fig. 4).

Using quality function deployment process, six key technical strategies are identified to deliver the customer's requirements. They include:

1) increase system availability,
2) decrease time of recovery,
3) reduce network congestion,
4) reduce application response time,
5) improve quantitative process control, and
6) improve quantitative quality planning.

Their trade-off relationships are then identified. For example, improvement in quantitative software process planning reinforces quantitative process

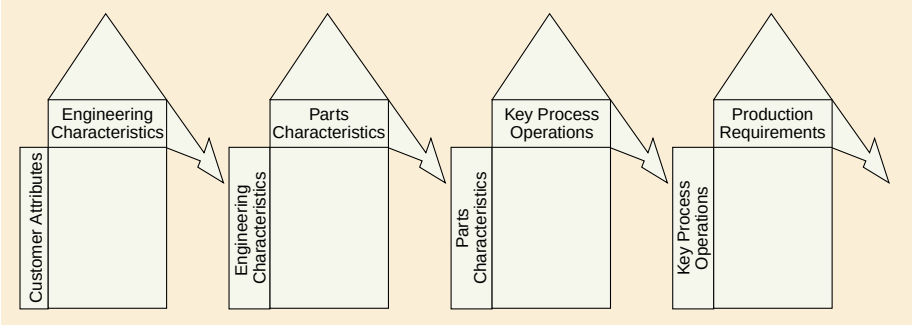

Fig. 2 A framework for quality function deployment

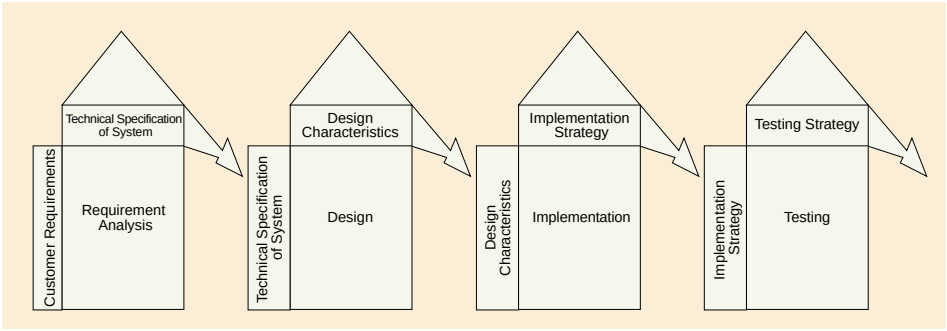

Fig. 3 A framework for software quality function deployment

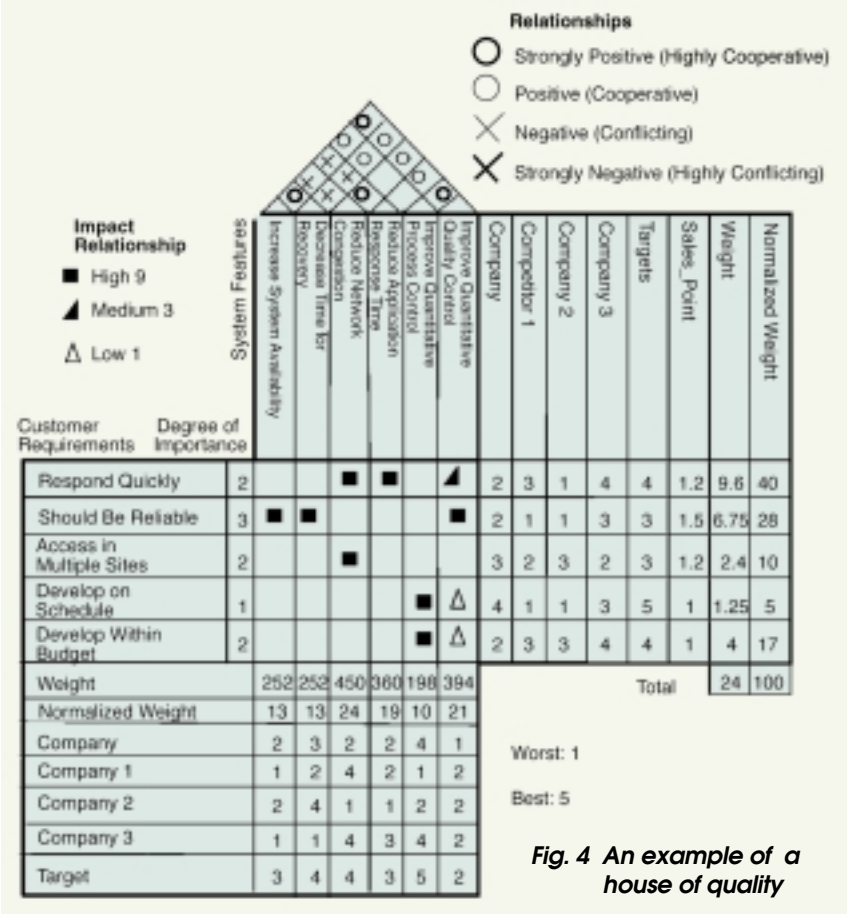

Fig. 4 An example of a house of quality

control. Therefore, they are positively related. The technical strategies are then related to customer requirements. Every customer requirement is covered at least by a technical strategy in this application, as shown in Fig. 4.

The competitive analysis in terms of both customer requirements and technical characteristics in the house of quality clearly shows weaknesses and strengths of the software product. For example, the company has the second lowest score in terms of satisfaction of the customer requirement "respond quickly." Its quality goal is set to be four (4), which is significantly higher than the current score of two (2). Two system features "reduce network congestion" and "reduce application response time" have strong positive impact on its satisfaction. We need to reduce network congestion and reduce application response time significantly in order to achieve the quality goal in terms of satisfaction of customer requirement "respond quickly." Therefore, target values of these two features are set to be significantly higher than their current ones in terms of their competitive assessment. Deployment of the system features can be developed further at the design and implementation level to achieve their targets. In addition, importance weights of both customer requirements and technical characteristics help us to allocate resources and assess their priorities. The technical strategies can serve as a basis of design for software evolution.

Summary

Software total quality management is widely applied in the development and the evolution of software products. Software quality function deployment can help to improve software quality, increase software productivity by avoiding rework, and decrease software development cost. In addition, it can be used to increase customer satisfaction by incorporating customer voices into software development and evolution activities in its life cycle.

Read more about it

• Akao, Y., *Quality Function Deployment*, Productivity Press, Cambridge, MA (1990).

• Erikkson, I., "Quality Function Deployment: A Tool to improve Software Quality," *Information and Software Technology*, 35(9), pp. 491-498 (1993).

• Haag, S., M.K. Raja and L.L. Schkade, "Quality Function Deployment Usage in Software Development," *Communications of the ACM*, 39(l), pp. 41-49 (1996).

• John R. Hauser and Don Clausing, "The House of Quality," *Harvard Business Review*, May-June issue, pp. 63-73 (1988).

• Lowell Jay Arthur, *Improve Software Quality—An inside's Guide to TQM*, John Wiley (1993).

• ZuItnerr, R.E., "Quality Function Deployment (QFD) fGr Software: Structured Requirements Exploration," in Schulmeyer, G.G. and J.I. McManus, ed., *Total Quality Management for Software*, Van Nostrand Reinhold, New York (1992).

About the author

Xiaoqing Frank Liu is currently an associate professor at the Computer Science Department in the Texas Tech University. He graduated with a Ph.D. in computer science from the Texas A&M University at College Station in 1995. He has conducted active research in the areas of software engineering, requirements analysis, software quality, database systems, and knowledge-based systems since 1985. He has published more than forty refereed journal and conference papers in the above areas.