Proceedings of ICAD2000
First International Conference on Axiomatic Design
Cambridge, MA – June 21-23, 2000
ICAD063

ADDRESSING CHANGING CUSTOMER NEEDS BY ADAPTING DESIGN REQUIREMENTS

Jason D. Hintersteiner
SVG Lithography Systems, Inc.
901 Ethan Allen Highway
Ridgefield, CT 06877
hintersj@svg.com

ABSTRACT

One of the most important aspects of the product development process is to develop an understanding of the true needs of the customer that must be satisfied by the design. While fine in principle, this understanding is very difficult to achieve in practice, as there is usually not a one-to-one correlation between the stated needs of the customer and the corresponding requirements that the design must satisfy. Accordingly, great effort must be made by product designers to translate the needs and desires of the customer into appropriate functional requirements and constraints for the design. Because customer needs often change during the product development cycle, the requirements of the design may change dynamically. For a design to be a success, therefore, it is vitally important for designers to understand the impact of changing customer needs on the design requirements.

A design enterprise can be modeled as interfaces between product development, manufacturing, suppliers, customers, and support/field servicing. In order for the enterprise to be successful, effective communication across these interfaces is essential. This paper, therefore, examines the interface between product development and the customer in detail, in order to better understand what information is and should be exchanged, so that customer satisfaction with the design process can be improved, especially in response to changing customer needs. Furthermore, a system design technique based on Axiomatic Design theory is discussed as a tool that can be used to improve communication between the customer and design engineers, once the initial design concept is established.

Keywords: Lean Manufacturing, Lean Enterprise Model, Customer Needs, Systems Engineering, Axiomatic Design

1 INTRODUCTION

In competitive manufacturing industries, there is a drive to provide customers with products of very high quality and within very short development and manufacturing times. Accordingly, there is a great deal of interest in applying lean manufacturing techniques to the design and manufacture of products. These techniques break away from the traditional compartmentalized outlook of the entire product life cycle, and instead try to maintain a more systematic perspective. Due to the complexity of large-scale design and manufacturing systems, however, implementing lean manufacturing techniques is not necessarily an easy task. It often requires massive corporate reorganization, and requires the support of all employees and interested parties in order to be successful. (Womack et al., 1991), (Womack & Jones, 1996)

Accordingly, research efforts are underway to investigate how lean manufacturing techniques can be more easily applied and adapted to existing corporate design and manufacturing systems. One such project is the MIT Lean Aircraft Initiative (LAI), which examines methods to reduce cost and cycle time across the full design and development cycle, while continuing to improve product performance. To accomplish this, the LAI has developed a framework for representing and understanding issues in lean enterprises, known as the Lean Enterprise Model (LEM). While the focus of this research is on military aircraft applications, the model is sufficiently general to apply to enterprises in other industries. One of the newest areas of research in the LEM is an examination of the interfaces between the major players in a manufacturing enterprise. The LEM is outlined in more detail in Section 2.

One of the key interfaces defined by the LEM is the link between customers and product development. This link is very important in any business enterprise, as the final product designs must address all of the needs of the customer. This paper focuses on the details of this interface, including an examination of the types of information that must be communicated across this link, as well as a discussion of the real and perceived impediments to communication.

Changes to the design specifications during the design process, usually resulting from real or perceived changes of the customers' needs, encompass the most important information that is exchanged between customers and product designers. These changes can often have a profound, and unfortunately detrimental, impact on the final quality of the design. In many high-tech industries, for example, product designs often push the frontier of technology, which usually results in long lead times for the design effort. Accordingly, customers may not accurately predict at the beginning of the design cycle the performance that will be required of the final product. Nevertheless, demanding customers will want the design to be completed with a minimum increase to the original price and a minimum delay to the original schedule, despite the fact that changes to the requirements can necessitate significant rework of the design.

To attempt to meet such cost and schedule constraints, there is usually an incentive to minimize the changes to an existing design. It is generally impractical to discard all of the design work that has already been done, especially for relatively late changes to the requirements. What this means, however, is that a system designed for one set of requirements is retrofitted to satisfy a new, different set of requirements, which generally results in a sub-optimal design solution. It is conceivable that, had the final set of design requirements been established initially, a different design path would have been followed, resulting in a significantly different design. While some changes may be simple to implement and not significantly impact other parts of the system, other changes will generally affect several parts of the design, often in unpredictable ways. This discussion is provided in Section 3.

Since companies want to satisfy their customers, changing customer requirements must be accepted as part of doing business. This does not mean, however, that changes to the design requirements should be accepted blindly – product designers must understand what impacts a proposed change will have on the established design. In order for customers and product designers to make informed choices as to which requirement changes are viable and practical, tools are required to highlight the design impact of such changes. One such tool, known as a system architecture, has been developed to capture all of the functional requirements, design parameters, and constraints of a design, along with their interrelationships. By incorporating Axiomatic Design principles, the system architecture evaluates the quality of a design and its robustness to changing requirements, as well as showing how a proposed design change impacts other aspects of the design. A review of Axiomatic Design principles and the system architecture is presented in Section 4, and an example of how the system architecture has been used to understand and track changing customer requirements is presented in Section 5 for the design of a commercial photolithography system.

2 THE LEAN ENTERPRISE MODEL

The Lean Enterprise Model (LEM) incorporates lean enterprise principles and practices, developed and supported by research-based benchmarking data derived from surveys, case studies, and other activities. The goal of the LEM is to evaluate an enterprise's current state of "leanness" in its organizations and processes, as well as to provide insights as to where lean efforts should be directed in the future. The LEM identifies the following overarching practices that are critical for successfully implementing lean enterprise techniques: (LAI, 1999)

1. Identify and optimize enterprise flow
2. Assure seamless information flow
3. Optimize capability and utilization of people.
4. Make decisions at the lowest possible level
5. Implement integrated product and process development
6. Develop relationships based on mutual trust and commitment
7. Maintain challenge of existing processes
8. Nurture a learning environment
9. Ensure process capability and maturation
10. Maximize stability in a changing environment
11. Continuously focus on the customer
12. Promote lean leadership at all levels

The LEM also highlights the need for information interfaces among the major players in a manufacturing system, mainly product development, manufacturing, suppliers, customers, and support/field service. The interface model is shown in Figure 1. This representation naturally shows the ideal situation and oversimplifies what occurs in actual practice – traditionally, while information has always been exchanged along these interfaces, these groups tend to run autonomously, and therefore have their own unique technological and managerial cultures. The LEM advocates that such traditional behavior impedes communication, and can have a detrimental effect on the quality and timeliness, and therefore the ultimate value, of the information exchanged.

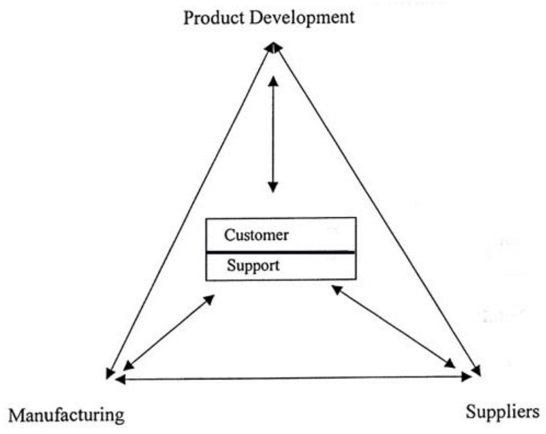

Figure 1: LEM interface model. (Courtesy Dr. Deborah Nightingale, LAI)

In order to make the communication links between each of these major players as simple and as open as possible, an understanding must be derived of the barriers and impediments to communication between these distinct groups. The key is to make the distinction between *real* and *perceived* communication constraints. Real constraints are legitimate impediments to communication, for which there may be no simple solution that is managerially and/or technologically feasible. In contrast, perceived constraints are the barriers that emerge from the history and culture of the enterprise, which can be overcome with sufficient time, effort, and dedication by all of the stakeholders. According to several case studies undertaken by Womack and Jones (1996), perceived barriers are often imposed by long-time workers and managers themselves, who have too much of their power and position invested in the established set of practices. New people often must be brought in who have the skill and insight to observe the enterprise from a fresh perspective. Unfortunately, distinguishing between real and perceived constraints is not a simple task, and the distinction is unique to each individual enterprise.

3 THE INTERFACE BETWEEN CUSTOMERS AND PRODUCT DEVELOPMENT

The first step in establishing good communication between customers and product designers is for a company to show willingness to nurture and develop a good relationship with its customers. A company's product development team must be responsive to the customer, in order to develop a good understanding of their needs and desires. According to Hagel and Singer (1999):

> Finding and developing a relationship with a customer usually requires a big investment. Profitability hinges on achieving economies of scope… extending the relationship for as long as possible and generating as much revenue as possible from it… [Therefore,] businesses naturally seek to offer a customer as many products and services as possible. It is often in their interests to create highly customized offerings to maximize sales. Their economic imperatives lead to an intently service-oriented culture. When a customer calls, people in these businesses seek to respond to the customer's needs above all else. They spend a lot of time interacting with customers, and they develop a sophisticated feel for customers' requirements and preferences, even at the individual level. (pp. 136)

In order to build a successful relationship with customers, a company must show that it is responsive to customer needs, makes products of high quality, can deliver products on-time, and acts in an ethical and reliable manner. (Ertas and Jones, 1993)

A customer's impression of a company will mainly emerge from how the company markets itself to sell its products, as well as the quality and reputation of a company's previous products. Ideally, a company would like to build a reputation for building high quality products while achieving on-time delivery, and then use that reputation to market future products. In order for this to work, however, a company's marketing department must not only show responsiveness to the customer, but must also develop a good understanding of the customer's needs. This is easier said than done.

3.1 MARKETING: THE MIDDLEMAN DEPARTMENT

The traditional role of the marketing department is to understand the so-called *voice* of the customer. In many commercial industries, such as consumer goods, there is little if any direct contact between customers and product designers, as it is the role and mission of the marketing department to understand the needs and desires of customers, and translate them into a set of design requirements for product developers. With respect to the LEM interface model, marketing lies along the interface between customers and product development. This is depicted graphically in Figure 2.

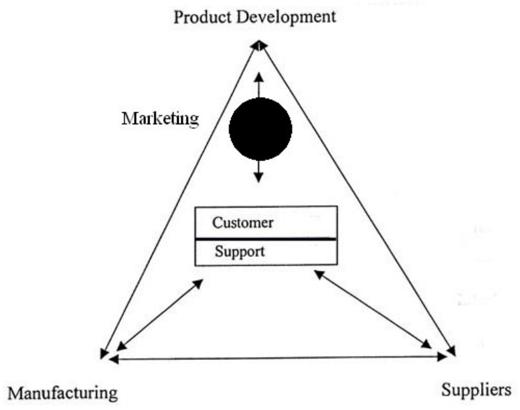

Figure 2: LEM interface model, modified to highlight marketing.

However, as any child who has ever played the game "telephone" can tell you, the more indirect the path of the information from its source to its final destination, the more likely that some information will be misinterpreted or lost along the way. How can this happen with marketing? Several factors can play a role. Marketing polls current and potential customers through "market research," which is a process where customers are interviewed regarding what they like and dislike about current products available, as well as what changes and features they might like to see in the future. Of course, the answers received depend on the questions asked. A marketing department that is not knowledgeable about the details of the product design process, for example, may not ask the right questions, or may not understand all of the implied ramifications that can emerge from certain answers.

Another more common problem emerges from the fact that different customers want different things. Ideally, marketing should undertake the effort to filter these options and develop a streamlined set of requirements. This, however, does not always happen in practice, and it is disturbingly easy for the requirements to become overspecified. This is not the wisest course of action from a technological or a financial point of view, as overspecification of the design requirements can result in unnecessary functionality, and therefore unnecessary complexity, in the design. According to Suh (1999), increased complexity increases the information content of a design, and therefore violates the intuitive and usually correct notion that the simplest design is ultimately the best in terms of cost and quality.

3.2 IDENTIFYING WHAT THE CUSTOMERS WANT

So what do customers want? Often, they do not know themselves. In fields where most customers are not technologically educated, such as consumer goods, customers may have neither the skills or background to express themselves in appropriate terms nor the knowledge to understand what is or is not feasible from a technological and financial point of view. Thus, the difficulty lies in understanding what the customers *actually* need, and not necessarily what they *say* they need. According to Slocum (1992):

> In addition to solving and identifying problems, the design engineer must also learn to identify what the customer really needs, which is not necessarily what the customer

thinks that he or she needs. This requires interaction with marketing research groups, customers, and manufacturing personnel on a continuing personal basis. (pp. 18)

Conversely, in high-technology fields, such as the semiconductor manufacturing industry, the customer has the problem of not necessarily knowing what their needs are going to be when the product is eventually finished. This is mainly due to a highly competitive environment which often changes at a faster pace than the product development cycle. As a result, customers must make an educated initial guess as to what their needs will be, and then refine their requirements during the product development cycle as their actual needs become more well-defined. Fortunately, the customers in such industries usually understand (and often drive) the technology involved, and thus direct communication between the designers and the customers is not only possible but encouraged. Indeed, such communication is essential in order for customers to ensure that their suppliers sufficiently understand their needs. Such direct communication, however, can also be impeded by one customer desiring to keep proprietary information from other customers. Accordingly, a customer may deliberately withhold some critical information from its suppliers to minimize the risk of that information being leaked, even accidentally, to their competitors.

A successful design enterprise must be able to understand what the customer really needs. This can be accomplished either by observing trends and predicting where markets will develop, or by creating a customer need where no such need existed before. In order to do either of these, the company must first understand who the customers are.

3.3 IDENTIFYING THE CUSTOMER

A customer has traditionally been perceived as the purchaser of the product. However, if the customer is a large company, the people making the decision to buy the product are rarely the same people who will actually *use* the product. The end-users are clearly "customers," even though they are not directly paying. Thus, the design must account for all of the customers of a product, and not just the needs of the purchaser.

A good product design must therefore consider how the product will be used by an operator. This information is not often readily available. Here, the term "operator" both refers to the end-users of the product and the people responsible for maintaining and servicing the product. Thus the design of user-friendly and ergonomic interfaces, as well as ease of serviceability, are extremely important. Research by Leveson (1995) has concluded that the overwhelming majority of incidents and accidents in large-scale systems tend to result from poorly specified design requirements. Among other observations, Leveson has noted that the design requirements frequently overlooked includes minimizing boredom in cases where repetitive tasks are necessary, considering involuntary reactions during crisis situations, and understanding potential ways that the system can be misused.

This is just one example of *unspecified requirements*, which are requirements on the design that were either considered to be relatively unimportant or that were inherently assumed as intuitive at the beginning of the design process but never explicitly documented. Such requirements tend to be either forgotten and/or de-emphasized during later stages of the design process. Unfortunately, these requirements don't tend to reemerge until either late in the development process, or sometimes even after the product is released. At this point, corrective action can be very expensive both in terms of financial cost as well as customer dissatisfaction. Accordingly, product designers must realize that not all of the requirements that must be satisfied are explicitly defined at the outset. It is the duty and responsibility of designers to consider all of the requirements for their design, even if some of those requirements are unstated and non-obvious.

3.4 MANAGING DESIGN TRADEOFFS

Unfortunately, it is not always possible to satisfy every requirement completely and successfully, as there are often inherent tradeoffs that must be reconciled. According to Leveson (1995):

> Desirable qualities tend to conflict with each other, and tradeoffs are necessary in any system design or development process. Attempts to design a system or a development process that satisfies all desirable goals, or to provide standards to ensure several goals without considering the potential conflicts, will result only in failure or in *de facto* (and non-optimal) decision making. (pp. 69-70)

Different stakeholders often want different design features and performances, necessitating the need for tradeoffs. The choice of which tradeoffs to make are often influenced by the fact that large customers are much more likely to be catered to than customers who provide very little business. This is not without its problems, however. A large customer is in a better position to demand more design features and changes that are tailored to its specific needs, which may not necessarily be compatible with industry standards and/or the needs of other customers in the marketplace. In order to mitigate such problems, many companies are looking into the concept of *mass customization*, which is a movement to develop products that incorporate enough flexibility in the design and manufacturing processes to allow the product to have several customizable features. Conceptually, running small batches of products with particular configurations can allow a company to cut down inventory while attracting a wider customer base, thereby decreasing overall costs while increasing revenue.

By definition, however, increased flexibility in the design increases the design's complexity, and thus the information content. Given the benefits of decreasing inventory and satisfying a wider customer-base, some increased design complexity can be justifiable. It is dangerous, however, to view greater flexibility as a panacea, since the product design can only accommodate the anticipated flexibility needed for reasonably predictable requirements. Good knowledge of current and future customer needs is therefore essential if the *correct* flexibility is to be designed into the product; otherwise, unnecessary functionality may be included and the overall quality of the design decreases.

Leveson (1995) refers to this problem as the *curse of flexibility*. While most of Leveson's research focuses on issues in software design, her conclusions are directly applicable to hardware and system designs. The desire to promote flexibility often encourages the designer to create what *can* be accomplished, rather than what *should* be accomplished.

> Flexibility also encourages the redefinition of tasks late in the development process in order to overcome deficiencies found in other parts of the system. During development of the C-17, for example—a project that has run into great difficulties largely because of software problems—the software was changed to cope with structural design errors in the aircraft wings that were discovered during wind tunnel tests. This case is typical… Discipline is also necessary [to limit] the functionality of the software. This discipline may be the most difficult of all to impose. Theoretically, a large number of tasks can be accomplished with software, and distinguishing between what *can* be done and what *should* be done is very difficult. Software projects often run into trouble because they try to do too much and end up accomplishing nothing… The flexibility of software… encourages us to build much more complex systems than we have the ability to engineer correctly. A common lament on projects that are in trouble is "If we had just stopped doing *x* and not tried to do more…" (pp. 34-36)

3.5 The Importance of Well-Specified Requirements

The discussion above serves to stress the importance of having well-specified requirements. The earlier in the design cycle that requirements can be accurately specified and clarified, the better off the overall design endeavor. According to Ertas and Jones (1993):

> If the requirements are too stringent, the project cost will escalate and (possibly) no supplier will be found that is willing to bid on the contract to provide the item in question. If the requirements are too lax, the overall system requirements may not be met, which could lead to dire consequences for the overall project. An additional problem with loose requirements is that they end up being tightened with greatly increased cost, difficulty, and ill will between the supplier and the customer. The importance of establishing valid design requirements is thus apparent… A good specification will minimize problems of interpretation that could surface later and result in disagreement with the supplier, possibly with negative impact on the entire project. (pp. 14-15)

Every time significant changes to the design requirements are made, corresponding delays and cost increases are likely to result. In reality, a customer's true needs are always changing, though at some point the decision must be made to complete the design and build the product. According to Ertas and Jones (1993):

> In most design processes of any significant magnitude, a design *freeze* is implemented at some point prior to completion. This is the point at which the design process is formalized and design changes are placed under strict and formal control, often by some sort of configuration control board, [which] normally include[s] membership representing all of the design disciplines, project management, the customer, safety, quality control, and other staff functions, as appropriate. The point in the overall design process at which the design is *frozen* is determined by customer requirements, by the need to control costs and configuration, by the need to inject greater discipline into the process, and by the need to forceably [sic] implement increased coordination among all the participants in the program. (pp. 19)

Often, it is advantageous for the product designers to negotiate changes to requirements and suggest modifications that the customer may agree with, in order to make the product development process easier. Such negotiations, however, can take time and effort. According to Slocum (1992):

> The customer will usually be glad to review modifications to their specifications in the hope that it will save them money; however, they will probably be wary that you are trying to sell them something from your existing inventory just to unload it. As a result, several iterations are likely before a final set of specifications is agreed upon. (pp. 29)

A reasonable set of requirements cannot be firmly established without understanding how the different parts of the design interact with one another. The next section discusses a tool which is useful for highlighting the ramifications that potential design decisions and alterations may have on the overall system performance, and therefore can give both customers and product developers a more rational and scientific basis to negotiate changes to the design requirements.

4 INTERFACE TOOL BETWEEN CUSTOMERS AND DESIGNERS: AN AXIOMATIC DESIGN SYSTEM ARCHITECTURE

When a system becomes sufficiently complex, the design process must be distributed among several engineering design teams, each of which is given smaller and more manageable tasks to accomplish. Typically, each design team will try to optimize its design based on its assigned tasks and constraints. It is very easy, however, for a designer to be unaware of "external" design decisions that can have a negative impact on the design. Thus, when all of the individual designs are put together, it is quite common to discover that the overall system does not function as intended. This results from the fact that each individual subsystem design has been locally optimized, without accounting for the complex design interrelationships that exist between the subsystems. Accordingly, a change to one portion of the design can negatively impact other portions unintentionally, because no framework was used to trace the impact of the design choices between the task divisions.

This can become especially problematic when the customer requirements change during the design effort. As previously mentioned, it is generally not feasible to restart the design process from scratch, so new and modified functional requirements and constraints must be incorporated into the existing design as the changes occur. The lack of a systematic framework to trace the impact of changing requirements and design decisions can then lead to a breakdown of proper communication between the design teams. One design team may not be aware of changes to another group's requirements, even though they are significantly impacted.

According to Söderman (1998), the main source of difficulty in designing large systems is that, in most cases, good representations of the design either do not exist or are not used to their full potential. Hence, it is extremely important to have a methodology to trace the impact of design decisions on both local and system-wide levels, since the real goal of the design effort is to optimize the performance of the *system*, which does not necessarily mean optimizing the performance of each *component*. Here, a system is distinguished from a component in the sense that a system not only consists of process functionality, but also requires distinct functionality for controlling and supporting the processes. By treating each level of the hierarchy as a system composed of subsystems, the same classes of functional requirements appear at every hierarchical level. Thus, the system is represented in a *recursive* manner so that, no matter how deep in the hierarchy one looks, the general pattern and layout of the representation remains consistent. This concept is shown graphically in Figure 3. Accordingly, a fractal representation emerges of the different subsystems that are incorporated into the hierarchy of the overall system – the details at each level are unique, but the overall types of functions and interrelationships remain consistent.

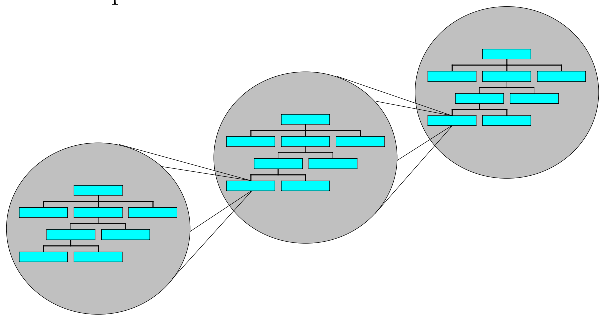

Figure 3: The concept of a fractal representation for systems. Each system is only one part of a larger system's hierarchy.

Given this definition, Axiomatic Design theory can be applied to the design of complex systems. This approach generates a *system architecture*, which captures the hierarchical structure of the functional requirements, design parameters, and constraints of a system. Using the design axioms, the quality of the design can be evaluated, so that potential problems of a particular design can be detected and addressed during the design process.

Accordingly, the system architecture can be used as a communication tool between groups of design engineers as well as between design engineers and customers. The system architecture highlights the relationships between the functional requirements, design parameters, and constraints, and can be used to evaluate the impact of proposed design changes as well as changing functional requirements and constraints. As a result, the system architecture can allow product designers and customers to understand the proposed changes and make more informed decisions as to whether or not such changes should be pursued. The system architecture is outlined in more detail below.

4.1 AXIOMATIC DESIGN

Design is defined as the development and selection of a means (design parameters, or DPs) to satisfy objectives (functional requirements, or FRs), subject to constraints (Cs). Axiomatic Design provides a framework for describing design objects which is consistent for all types of design problems and at all levels of detail. Thus, different designers, as well as observers to the design process, can quickly understand the relationships between the intended functions of an object and the means by which they are achieved. Additionally, the design axioms provide a rational means for evaluating the quality of proposed designs, and guides designers to consider alternatives at all levels of detail by making choices among these alternatives more explicit. The main concepts of Axiomatic Design are: (1) *domains*, which separate the functional and physical parts of the design; (2) *hierarchies,* which categorize the progress of a design in the functional and physical domains from a systemic level to more detailed levels; (3) *zigzagging*, which indicates that decisions made at one level of the hierarchy affect the problem statement at lower levels; and (4) *design axioms*, which dictate that the independence of the functional requirements must be maintained and that the information content (i.e., cost, complexity, etc.) must be minimized in order to generate high quality designs. More thorough explanations and detailed case-study examples of Axiomatic Design theory are available in (Suh, 1990), (Suh, 1999), and (Tate, 1999).

4.2 SYSTEM ARCHITECTURE TEMPLATE

For large systems, a system architecture is developed to break down the design into individual subsystems at each level of the design hierarchy. In this representation, a system is modeled as a series of interacting inputs and outputs. These inputs and outputs follow a template at every level of the design hierarchy, and fall in to one of the following functional categories: (1) *process functions*, the elements of the system that perform value-added tasks and activities; (2) *command and control functions*, the logic that schedules and coordinates the process functions; and (3) *support and integration functions*, the support structure for the process and control functions, including pneumatics, mechanical structure, electronics, etc. (Hintersteiner, 1999)

Most complex electromechanical systems consist of both hardware and software. Unfortunately, hardware engineering and software engineering are all too frequently treated as separate disciplines. It is therefore not uncommon to treat these tasks separately, with independent groups of designers, in product development. Since the design of the software control logic depends on what hardware must be controlled, software is typically developed after the hardware is mostly, if not completely, defined.

According to Leveson (1995), this dependency leads to very short development times for software, as well as the last-minute addition of functionality to the software because of either hardware limitations or the perception that it will be "easier" to implement certain tasks in software. While software is more flexible than dedicated hardware, this philosophy often leads to undue software complexity. As a result, the software can behave unpredictably, since it is impossible to test software under all possible operating conditions. This can lead to sub-optimal software design, especially in terms of the quality of the user interface and in the satisfaction of unspecified requirements which emerge late in the design process. Recent efforts in concurrent engineering have been made to perform more hardware and software design in parallel in order to reduce overall development time. This cannot be done effectively, however, unless the intended functional requirements of both the hardware and the software are well understood from the outset, and their interrelationships can be identified. Accordingly, tools are needed to document the interrelationships between the hardware and software in a system.

In the system architecture, the software in electromechanical systems is represented by means of command and control algorithms (CCAs) at every level of the design hierarchy. At each level, the CCA captures the logic of the interactions among the hardware elements at that level, along with all of the communication protocols necessary to interact with its immediate parent and child CCAs. Thus, a software hierarchy emerges that mirrors the hardware hierarchy, and the software design is embedded within the hardware design (Hintersteiner & Tate, 1998).

With minor changes to the terminology, the system architecture template is applicable to the design of the control software in complex systems. The software programs used to control systems satisfy key value-added tasks that are necessary for the entire system to run effectively. However, the software programs alone are not sufficient to provide control – the order in which the tasks are performed, the need to interface with users (i.e., operators, maintenance engineers, or autonomous external computers), and the need to recover from errors and provide other supporting tasks are also important to direct and support the overall goal of providing the control functionality. Thus, it is important for the design of the CCA to capture all of these elements. Accordingly, it is necessary to consider each CCA as a system in its own right. (Hintersteiner & Nain, 1999)

4.3 CONSTRAINTS

In addition to identifying the relationships between the FRs and DPs in hardware/software systems, the impact of constraints on the FRs must be well understood. Constraints are defined as the set of performance specifications and design restrictions that impact the FRs and therefore limit the range of acceptable design solutions (DPs). In some sense, they ask "how well" specific tasks need to be performed. Since the system architecture organizes the design in terms of its functionality, it can be used to determine how customer and management constraints impact the FRs and therefore restrict the scope of potential design solutions. The performance specifications provided by the customer, management, government and industry standards, and safety regulations are specified as constraints at the system level. Like FRs and DPs, constraints are refined and clarified as decomposition progresses. Many high level constraints also influence the specification of lower-level FRs. In general, the more constraints that exist at the system level and the more restrictive the constraints (e.g., tighter tolerances), the harder it will be to generate an acceptable design solution. Accordingly, the harder it will be to maintain functional independence while minimizing the information content.

In order to properly identify the importance and stringency of various constraints at particular levels of the hierarchy, it is useful to categorize the constraints. These classifications appear at every level of the system hierarchy, maintaining the fractal representation of the system architecture. (Hintersteiner, 1999)

4.3.1 Critical Performance Specifications

This is the set of constraints that are most critical for the system to be considered a success, and incorporate the metrics by which a system will be judged by its customers. Generally, these constraints will have specific values or impose specific tolerances that must be achieved or exceeded in order for the system to be acceptable to the customer. Examples of this type of constraint include throughput specifications, reliability requirements, and specific process performance metrics.

4.3.2 Interface Constraints

This is the set of constraints that describe how the system must interact with its environment. This includes specification of all interfaces between the system and the environment, including usability by operators and maintenance personnel, as well as specific features or options that particular customers may desire (e.g., a custom layout so that the system will fit into a preexisting facility). Generally, these constraints include the types of operands / inputs that the system must handle, as well as specific parameters and features that must be incorporated into the design of particular components. This category also includes constraints on human and equipment safety, as well as acceptable impacts on the environment.

Several interface constraints tend to emerge from choices and tradeoffs made elsewhere in the design of the system, including acceptable factors of risk. For example, the choice to use a particular robot for an application may lead to limitations on where that robot can reach and, therefore, restrictions as to where accessible stations need to be placed. In addition, vibrations induced by the robot may dictate requirements for vibration isolation of other components in the system. If a different robot or another type of mechanism is selected, such derived requirements may change or become unnecessary.

4.3.3 Project Constraints

This is the set of constraints dictated by marketing and management that impact the resources and time available for the design effort. Thus, this category incorporates such items as the schedule for delivery, design reviews and other project deadlines, specific test procedures, development budgets, and staffing and resource limitations.

5 CASE STUDY EXAMPLE: RETICLE MANAGEMENT SYSTEM

The system architecture template presented above has been applied to several industrial design projects, most notably in the designs for photolithography tools manufactured by Silicon Valley Group, Inc. Lithography exposure tools are the key to advanced semiconductor device fabrication, with technical advances in this field serving as the critical technology in leading-edge device designs. Photolithography is a process whereby a desired pattern is optically transferred from a master onto the surface of a wafer coated with photoresist (a compound that is chemically sensitive to particular wavelengths of light). Step-and-scan photolithography divides the wafer surface into duplicate fields, where the master field is contained on a chrome-coated glass plate called a reticle. For every field, the wafer is stepped to the beginning of the field, and then a light beam is scanned across the reticle, exposing the field on the photoresist-coated wafer and thereby imprinting the reticle pattern into the field. A schematic diagram of the photolithography process is shown in Figure 4.

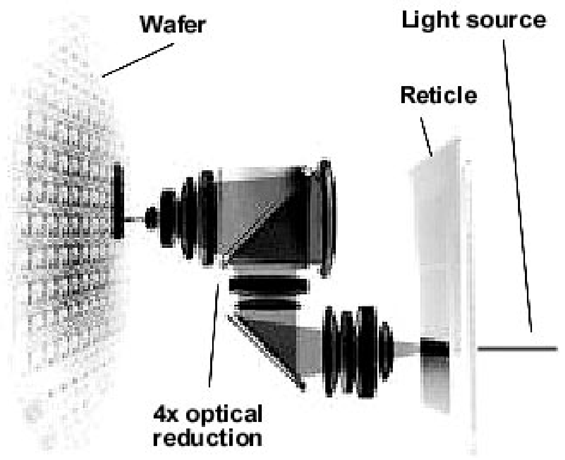

Figure 4: Schematic diagram of the photolithography process.

Photolithography integrates several different scientific and engineering disciplines, including optics, lasers, mechanics, dynamics, mechatronics, fluids, heat transfer, controls, and software design. In order to achieve the desired performance constraints, tolerances are extremely tight. Furthermore, products tend to be evolutionary, as short development times dictate the need to reuse as much of the existing design as possible. (The typical product cycle for these tools is 18 months.)

Accordingly, most development effort is usually devoted to refining the performance of existing subsystem designs. This is generally done because designing a new subsystem from scratch is perceived to cost more in terms of time, money, and manpower. This perception, however, can be very deceptive, as the refinements, though minor in and of themselves, are cumulative, and tend to impact not only the inner workings of that subsystem, but also its interfaces to other subsystems.

Thus, making initial changes to a subsystem design concept will indeed only have a minor impact on cost. However, as the design is modified further over successive product generations, generally by different designers, layers of changes are introduced which continue to deviate from the initial functional intent. Furthermore, as the internal workings of the subsystem change, the material and information interfaces also tend to adapt, necessitating design changes to other subsystems. Hence, the more refinements and changes that are introduced, the harder it becomes to get the hybrid system to work and keep it working over time. Eventually, the customer constraints imposed reach limitations that cannot be addressed cost-effectively by continued refinements, and at this point a new technology platform must be adopted.

One example of this trend comes in the design of the reticle management system (RMS). The key process functional requirements of the RMS are to transfer reticles in and out of the tool, store reticles internally when not in use, and exchange two reticles at the appropriate station (the reticle stage) when a new active reticle is desired.

The old RMS design, which had served our customers well throughout several product generations, satisfied these functional requirements in the following manner. For reticle transfers, the reticles are transported to and from the tool in special carriers (known as SVG cassettes). The cassettes, each holding one reticle, were stored in an internal library mechanism. For reticle exchanges, a 2-DOF mechanism called a "long stroke" removed the active reticle from the reticle stage, stored it in the appropriate cassette in the library, indexed the library to the cassette containing the new reticle, and placed the new reticle on the reticle stage.

Customers imposed further constraints on these functional requirements, which in turn impacted the final design. For example, customers wanted the ability to exchange two reticles quickly, as some lithography processes require the use of multiple reticles on a single wafer. To perform this quick exchange between two reticles, a second 1-DOF mechanism (known as a short stroke) was included in the design to remove, hold, and place the second reticle, working in conjunction with the long stroke. A schematic diagram of the old RMS design is shown in Figure 5.

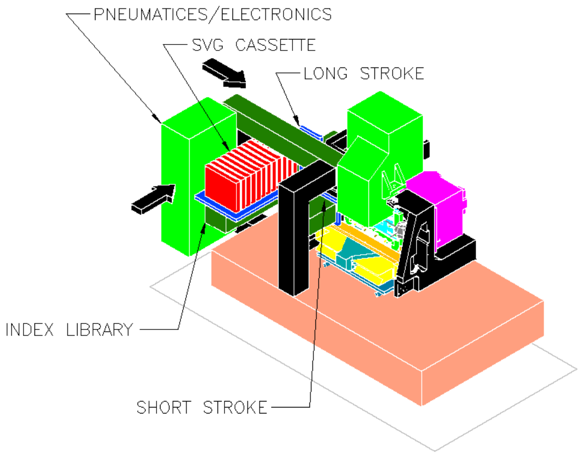

Figure 5: Schematic diagram of the old RMS.

While this design met the stated customer needs for several years with only minor modifications, it proved to be incompatible with newer customer constraints. Three constraints in particular prompted a complete redesign of the RMS, as outlined below.

(1) *Quick exchange of more than two reticles (process constraint)* – Some customers use more than two reticles during a process run. With the existing design, the short stroke could only be unloaded by loading the reticle back to the reticle stage. Thus in a three reticle process, for example, there would be a throughput hit if the second and third reticles needed to be exchanged, as the short stroke was occupied with the first reticle.

(2) *Reticles on demand (interface constraint)* – The existing design has the reticle cassette library inside the tool enclosure. In order to load / unload reticle cassettes, the tool itself must be opened, which requires any ongoing exposure processes to stop due to safety constraints. Thus, reticles can only be loaded and unloaded during tool servicing or between process runs. While this is not an issue in fabs using long production runs where the reticles are stored in tools and are

changed on the order of weeks or months, it is inadequate for fabs doing short production runs, since they often change the active set of reticles several times a day.

(3) *Mini-environment (interface constraint)* – Photolithography tools have conventionally been installed in Class 1 clean rooms, in order to prevent contamination of wafers and reticles as they are transported in the fab. (The inside of a photolithography tool is maintained at Class 1 by an internal environmental control system). Such rooms are very expensive, and so customers would like to place the tool in less expensive clean rooms (such as Class 100), and then transport wafers and reticles in mini-environment pods. The existing SVG cassettes are not hermetically sealed, and accessing the existing library requires the tool to be opened. Thus, placing the tool in a Class 100 clean room would result in reticle contamination.

Thus, while the essential process functional requirements of the RMS (namely transfer, store, and exchange reticles) remain fixed, the changing constraints dictated by customers imposed several changes to the sub-requirements, necessitating major revisions to the design. While it was technically possible to retrofit the old design to meet these new constraints, a preliminary analysis showed that retrofitting the old design would cost more in terms of time, money, and system reliability (and thus customer satisfaction) than it would to design a completely new system.

Accordingly, a new RMS has been developed, based on a different technology platform. For reticle transfers, the reticles are supplied to the tool by means of mini-environment pods which are placed on external indexers that are accessible without requiring the tool to be opened. The pods, along with an additional internal pod, are also used to store the reticles inside the tool. For exchange, a 1-DOF short stroke mechanism is used to remove reticles from the reticle stage, and a 6-DOF robot is used to retrieve a reticle from the pod library, place the reticle on the reticle stage, retrieve the old reticle from the short stroke, and place the old reticle back in the pod library. Thus, any two reticles (and not just the same two reticles) can be exchanged in a short amount of time. A schematic diagram of the new RMS is shown in Figure 6.

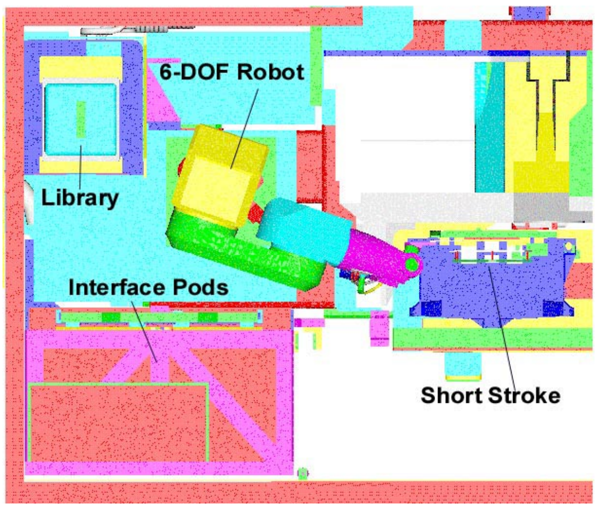

Figure 6: Schematic diagram of the new RMS.

Naturally, changes to these process FRs and the corresponding hardware design prompted new requirements for the control system and support systems associated with the RMS at every level of the design hierarchy. These systems were also subject to project constraints requiring that the new systems maintain compatibility with the rest of the existing tool. Thus, in addition to the hardware redesign, the control logic and support systems were also restructured.

6 CONCLUSIONS

The successful interaction between customers and product development is of key importance for success in a competitive marketplace. Customers and product developers must work together as partners in order to ensure that the design of the system will successfully address customer needs. In order to accomplish this, a company must build strong and mutually beneficial relationships with its customers. In order to maintain a successful interaction, the role of marketing must be defined, so that all of the customers, and their true needs, can be identified. Only by doing this can well-specified design requirements emerge, so that appropriate and intelligent design tradeoffs can be made.

To manage all of this, tools are needed to facilitate communication and the sharing of design ideas. One such tool presented in this paper is the system architecture. Based on Axiomatic Design techniques, this tool captures the functional requirements, design parameters, and constraints of the design, along with the interrelationships between them. As shown in the design of the robotic reticle management system, the system architecture can be used to translate changing customer needs into revised functional requirements and constraints. Thus, it provides a methodology for making logical design tradeoffs while maintaining an appropriate level of design flexibility to facilitate product evolution.

7 ACKNOWLEDGMENTS

The author would like to acknowledge the many people at SVG Lithography Systems, Inc., including J. Laganza, G. Friedman, P. DelMastro and the rest of the RMS development team, that provided the support and input necessary to make this effort possible.

In addition, the author would like to acknowledge Professor Deborah Nightingale at the Massachusetts Institute of Technology for her comments and insight into lean manufacturing techniques and the lean enterprise model (LEM).

8 ABOUT THE AUTHOR

Jason D. Hintersteiner received a Bachelor of Science and Master of Science in Mechanical Engineering at the Massachusetts Institute of Technology. His experience includes research projects in robotics, digital control, error modeling and compensation, computer networking, as well as two years of postgraduate work with Professor Nam P. Suh at MIT on the application of Axiomatic Design to large-scale systems. He is currently a Senior Staff Engineer at SVG Lithography Systems, Inc., and is chiefly responsible for providing systems engineering support and coordinating the implementation of Axiomatic Design throughout the engineering organization.

9 REFERENCES

[Ertas and Jones, 1993] Ertas, A. and Jones, J. C. *The Engineering Design Process*. John Wiley & Sons, Inc. New York. Copyright 1993. ISBN #047-151796-8.

[Hagel and Singer, 1999] Hagel, J. III and Singer, M. "Unbuilding the Corporation." Harvard Business Review, March-April 1999. Pp. 133-141.

[Hintersteiner, 1999] Hintersteiner, J. D. "A Fractal Representation for Systems." *Proceedings of the 1999 International CIRP Design Seminar*, Enschede, the Netherlands, March 24-26, 1999.

[Hintersteiner and Nain, 1999] Hintersteiner, J. D. and Nain, A. "Integrating Software into Systems: An Axiomatic Design Approach." *Proceedings of the 3rd International Conference on Engineering Design and Automation*, Vancouver, B. C. Canada. August 1-4, 1999.

[Hintersteiner and Tate, 1998] Hintersteiner, J. D. and Tate, D. "Command and Control in Axiomatic Design Theory: Its Role and Placement in the System Architecture." *Proceedings of the 2nd International Conference on Engineering Design and Automation*, Maui, Hawaii USA, August 9-12, 1998.

[LAI, 1999] MIT Lean Aircraft Initiative. *MIT Lean Aircraft Initiative Web Site*: http://lean.mit.edu/

[Leveson, 1995] Leveson, N. G. *Safeware: System Safety and Computers*. Addison-Wesley Publishing Co. Reading, MA. Copyright 1995. ISBN #020-111972-2.

[Slocum, 1992] Slocum, A. H. *Precision Machine Design*. Prentice-Hall, Inc. NJ. Copyright 1992. ISBN #013-690918-3.

[Söderman, 1998] Söderman, M. "Tools for Creating Understanding and an Integrated Dialogue in the Early Stages of Product Design." *Proceedings of the 2nd International Conference on Engineering Design and Automation,* Maui, HI. August 9 – 12, 1998.

[Suh, 1990] Suh, N. P. *The Principles of Design*, Oxford University Press, New York. Copyright 1990. ISBN #019-504345-6.

[Suh, 2000] Suh, N. P. *Axiomatic Design: Advances and Applications*. To be published by Oxford University Press, NY. Copyright 2000.

[Womack et al, 1991] Womack, J. P., Jones, D. T., and Roos, D. *The Machine that Changed the World: How Japan's Secret Weapon in the Global Auto Wars will Revolutionize Western Industry*. Harper Perennial, NY. Copyright 1991. ISBN #006-097417-6.

[Womack and Jones, 1996] Womack, J. P. and Jones, D. T. *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*. Simon & Schuster, NY. Copyright 1996. ISBN #068-481035-2.