Want Human-Centered Development? Reorganize the Company

by Donald A. Norman

What do you want for your product?
Good quality? Inexpensive? Quick to get to the market?
Good, cheap, quick: pick any two.

(Old engineer's saying.)

Development is a series of tradeoffs, often with incompatible constraints. Multiple factors compete for attention, each factor often demanding a solution that is incompatible with that required by another factor. Marketing, engineering, usability experts all champion their favored approach, each correct in their assessment, but nonetheless, each voicing different and incompatible concerns. How do these inconsistencies and incompatibilities get resolved? That is what the product process is all about: tradeoffs.

In high-technology companies, the problem is exacerbated by the fact that success in the early stages of the technology marketplace favors technology-centered, feature-driven products. Customers clamor for more and better technology: engineers become experts at providing a stream of continual improvements in power, increased features, all at decreased cost. In this world, engineering rules the show. Engineers reluctantly cede a place for marketing, and the reluctance is quite visible. Marketing, moreover, becomes primarily feature-driven: query the existing customers for the features they desire most and pressure the engineering team to add them to the product, often with little regard, understanding, or even interest upon the impact on the coherence and integrity of the product. Mind you, this feature-driven emphasis is probably correct: early adopters are not too concerned about the total user experience. Coherence in the product is of secondary importance, if it is noticed at all. These are technology-driven customers, customers who purchase their products based upon technological accomplishments, upon novelty and lists of features. This feature-driven marketing emphasis is quite appropriate at this stage of the product life cycle.

In the latter stages of a technology, the game changes considerably. As we saw in Chapter 2, as the technology matures, it becomes less and less relevant. The technology is taken for granted. Now, new customers enter the marketplace, customers who are not captivated by technology, but who instead want reliability, convenience, no fuss or bother, and low cost, both of original purchase and of upkeep. Factors such as the total user experience play a major role: customers want convenience and lack of hassle. This new entry, user experience, is not well established. Nobody quite knows how to deal with it.

The engineering team thinks it already understands user experience. After all, their previous customers were happy. The engineers themselves have no trouble with the product. Who are these new customers who need so much hand-holding? What's the matter with them, anyway.

The marketing group thinks it already understands user experience. After all, marketing is in close touch with the customer: it knows first-hand what they want. Do they want ease of use? Sure, add it to the list of features. Do they want an attractive product, sure, hire a graphics designer to make it look pretty. Each item gets added to the list of things to be accomplished, as if the total user experience were a feature like "more speed" or "more memory" that can be purchased or added on to an established design.

This assumption that user experience is just another add-on is pretty consistent across the industry. Consider how the problem of ease-of-use is handled. The assumption is that ease-of-use comes late in the game: after all, how can you make a product easy to use before it has been built? First we build it, say the engineers, then we bring in those user interface folks to add some graphics and menus and make it easy to use. The same goes for the technical writers: how can you describe how to use a product until it is all finished, so there is actually something to write about? The writer's job comes at the end.

The traditional sequence of product design is technology-driven. Marketing provides a list of essential features: the engineers state what neat new technical tricks and tools they are ready to deploy. The engineers build the device, putting as many new technologies to work as they can within their allotted time and budget, squabbling with marketing along the

way over which of those features really matter and which don't. Then after all is finished and the product ready to ship, call in the technical writers to explain it to the customers. Call in the graphics and industrial designers to make it look pretty. Call in the user interface experts to make it usable.

Guess what: this process doesn't work. For proof, we simply have to look around us at those high-technology products. For proof we simply have to ask why so many telephone help lines are required, why so many lengthy, expensive phone calls to use the product. For proof we simply have to go to the bookstore and look at how many bookshelves are filled with books trying to explain how to work the devices. We don't see shelves of books on how to use television sets, telephones, refrigerators or washing machines. Why should we for computer-based applications?

The high-technology industry is at a crossroads. The basic technology is now good enough for most purposes. It is at the transition point, in the gap between a technology-driven marketplace and a consumer-driven one. The customers have already made the transition to consumer products: they want convenience, they want simplicity. The high-technology companies, however, have not crossed over the chasm: they are still mired in technology driven product development, feature-driven marketing.

It is difficult for a company to make the transition to a consumer-driven marketplace. It requires an entirely new approach to products, it requires starting with an understanding of customers and their needs, their real needs, not the feature sets so loved by marketing. It means using social scientists to collect, analyze and work with the data, moreover social scientists on an even footing with engineers and technologists. It means structuring the whole product process differently than before. It may mean reorganizing the company.

In Chapter 9 I suggested that development start with the study of the true needs of customers determined through observation and structured interaction. Start with observations using psychologists and anthropologists, skilled social scientists who know how to observe and learn without disturbing the phenomena they are there to record. Analyze what has been observed. Do quick tests of design concepts. Do rough, quick mockups and try them out in the context of the customer's location, be it home, office, school or athletic field. Try different industrial designs. Try different interaction models. Then write a simple manual. Test as you go along on different populations, all representative of the target customer. Then, and only then, start with the technological factors and the design details.

"Start with psychologists and social scientists instead of technologists?" I can hear the rumblings throughout the company, "Hey: bringing in marketing was bad enough, but who are these people? What's happening to our company?"

The remedies I propose are not easy to implement: they call for restructuring of the product process. They call for changing the company culture. The hardest part of a company to change is its culture.

A company's culture is that nebulous, ill-definable set of knowledge, history, folklore, and spirit that determines how it goes about its daily activities, how it treats its employees, customers, partners and competitors. It is seldom written down, although written mission statements and company policy procedures are a major part of the culture. Company culture is often established in the early days of a company by its founders, then nourished until it establishes the deep roots of commitment that are difficult to see, to describe, and to change.

The Structure of the Product Team

Today, in the high-technology industry most existing products are revised once or twice a year driven by a marketing analysis of the competition and the features requested by the buyers and customers. The result is products that are technology-driven, feature-driven, defined by lists of critical features: items that are necessary, items that are desirable. The engineering teams struggle to get them into the product, and when schedule deadlines loom, a vast cutting exercise ensues in which the desired but not essential features are cut.

Lists of features completely miss the interconnected nature of tasks. Any given feature is apt to be insignificant in isolation, but critical when viewed as a whole, or perhaps seemingly important and critical when viewed in isolation, but prove to be insignificant, perhaps unnecessary when viewed in context. Ranking features is fundamentally wrong.

The proper way to make design tradeoffs is to understand why each of the various elements of a system exist, what roles they serve for the customer, or marketing, or technological performance. Then the tradeoffs can be made, pitting each element of the system against the costs in usability, marketability or dollars and schedule. When items are to be cut or prioritized, it must be done with a good understanding of the gains and losses from an overall point of view: looking at each item in context. A system-wide approach of the design in full context of its use, sale, marketing, and construction requires a new kind of product process, one that starts with the user and ends with the technology. And if this is taken seriously, it means a very different structure to the company.

The Design Police

Companies that are having difficulties with their products sometimes resort to heavy-handed bureaucracy: Is the design flawed? Are the products too difficult to use? Are service calls and help-line support rising? Call in the design police.

Years ago, other disciplines faced the very same problems. It wasn't long ago that marketing was also scorned, that it was thought to be something tacked on, after the fact. In a similar way, quality control was assumed to require no more than careful design and manufacture, once again, something that could be tacked on after the fact. Both cases proved wrong, but there is a lot to be learned by studying how marketing and quality control centers restructured the product process.

The same lessons that applied to marketing and quality control apply to user experience. Leave out marketing from the product conception, and the wrong products get built. Leave out quality considerations and no amount of testing will bring quality back in. Leave user experience out of the initial development constraints and by the time they are ready to be considered, it is too late: the die is cast. The user experience team becomes the design police. Calling in the police after the crime has been committed is not the proper way to go about things.

Calling in the police is a standard approach to remedying any flaw: legislate and police. None of these solutions work very well. The only way to get real, long-lived improvement is to attack the root cause of the problem. Sometimes this means better training, usually it means changing the reward structure of the company or of society. In the case of good design, it means both of these things, plus one more: reorganize the company.

The first step in undertaking a cure for bad design is to determine the underlying causes. After all, poor design is not deliberate. The reasons are manyfold, including ignorance of the principles of good development, lack of understanding of the people for whom the product is being designed, occasionally a lack of empathy. But even when these factors are eliminated, the organizational structure of the company often prevents good development practices from coming into play.

An abbreviated account of the history of reliability and quality of products is instructive. Quality used to be achieved by hit-or-miss. Employees were urged to work harder, to produce better products, without deviating from their assigned target rates of production, of course. This approach works no better than a parent telling a child to "be careful."

Nobody intentionally is careless in the conduct of their daily activities. Nobody deliberately sets out to produce a low quality product. Nobody wants their products to be difficult to use. These things happen because they are not the primary focus of activity. Our goals are to get to our next appointment, to finish our assignments, to get the job finished, to enjoy ourselves. Quality, ease of use, even personal safety, are the results of a process. These goals happen only if there is attention to the process that leads to unsafe practices, to low quality, and to poor design. Only after the processes have been discovered and new ones put into practice can one hope to see any changes. And these new processes are neither easy to develop nor easy to implement.

In the field of quality, the next step after exhortation was testing: continual testing of the product as it went through the manufacturing process. This had several virtues. It allowed defects to be discovered before the product was shipped. And it allowed for quantitative assessment of the problem, letting proposed solutions be tested against the requirement of true improvement in the measured results instead of the intuitive assessment practiced earlier. Enforcement of quality through testing is a clear case of design by legislation, with the dreaded quality police looking over your shoulder. It's a bad way to proceed and, although it can reduce incidents of failure and improve quality, it is seldom sufficient to produce true products of quality.

Once upon a time I was a reliability engineer. During my college years I had a summer job at Univac, then a major computer manufacturer. My job was to determine how to test circuit components for reliability. Engineers would design circuits, both their electronics and the printed circuit board layout. A few prototypes would be made and then submitted for approval by the reliability group. Note those words: "then submitted for approval." It was our job to approve the work that had already been done, not to find problems.

Not only were the electronic circuits submitted to us after all the work had been done, but by that time, the work was always behind schedule and probably above cost. If we were to say "no," everyone would be angry at us: the engineers, the managers, the company administration. Redoing the circuits would take time. Sometimes it would not even be possible, for other circuits had also been constructed that depended upon the configuration of the one we were supposed to approve. If we changed the physical dimensions, the pin-layout, or almost anything, the whole product would have to be redesigned. This was a product that cost a million dollars, back in the days when a million was real money. In other words, we were supposed to approve everything. Things had to be pretty bad for us to deny approval and get away with it.

Guess what? This method of getting quality doesn't work. The reward schedule of the company was wrong. We were not rewarded for doing our job, in fact, we were punished. Holding back approval delayed schedules and cost the company extra money. The engineers were not rewarded for attention to quality: after all, that was our job. The whole process of testing for quality at the end, coupled with the reward structure of the company guaranteed the failure of this method.

The proper way to get an acceptable product is through a cooperative process with all the concerned parties participating at every step of the way. If a company wants its engineers to design reliable circuits, the proper way is to teach them the principles of reliability engineering so that the principles are embedded throughout the development process. Let the reliability specialists serve as consultants, playing a role from the very start of the development and available to help when complex development problems arise. This transforms the reliability experts from the role of "dreaded police," a role that nobody enjoys, least of all the police themselves, to that of being trusted colleague, advisor, and co-worker. Why are we surprised when the result is superior? Reward the engineers for quality improvements. If service calls or the need for replacement parts decrease, reward the development engineers. Ask the development engineers to spend time with service representatives, with maintenance, with customers, to understand the way their products get used, the kinds of problems that exist. Make quality everyone's job.

Over and over again, experience shows that the only way to satisfy the many complex factors that must be considered throughout a project is through cooperative, interactive and iterative development where all interested parties play a role. Development is a series of tradeoffs. There is no correct design — the whole process is a series of compromises among a large number of conflicting requirements. Aesthetics, cost, usability, functionality, ease of manufacturing, ease of maintenance, ease of use, reliability, size, weight — all are legitimate, important factors, often at odds with one another. Make it less expensive and reliability might decrease. Add all the features customers require and usability and reliability might decrease while cost increases. Provide a portable system with the power and battery life customers demand and the weight and cost will increase, just the opposite of what the customers have requested. On and on it goes.

The traditional method of development is of a linear chain, from specifications, through design, manufacture, sales and delivery, usage and then repair, service, and assistance. Marketing provides the specifications, the engineers design to them, then pass the design on to the manufacturing experts. Somewhere along the way, the industrial design and human-centered development team is asked to do their jobs, and when all is finished, the technical writers are called in to write a cohesive, intelligible explanation and user manual. Finally, the sales teams are asked to sell the finished package and the service and maintenance people trained on its operation and on the kinds of problems that are expected. This is a time-tested method of development. It has led us to the complex, messy, unsatisfactory situation we are in today, especially in high-technology products, where customer satisfaction is low, the cost of help lines and service high.

How else can we proceed? Have all the affected parties participate together in the development process. This is a very foreign process to most engineers. But this is how to get human-centered development, this is how many companies have gotten quality, ease of manufacturing, ease of installation, ease of use, and ease of maintenance into their products. The morale is simple: Don't test it in, design it in. And the only way to do so, is to ensure that each of the experts in the many different disciplines that are relevant to the end product have a say throughout the entire product process, from conception to final delivery.

Once again, the story of total quality control is instructive. Today, many companies have completely restructured their development and manufacturing process, the better to make products of high quality. The process was often painful, taking years, changing organizational structure and, most difficult of all, changing the engineering and market development culture. But once everyone put quality as a high-level development goal, once it was realized that quality experts should be thought of as consultants and colleagues, the overall process was simplified. In the end, producing higher quality products through design and attention to the issues all the way through the development and manufacturing process ends up being cheaper and faster than the old way of doing business.

So too will it be with the other issues of concern to companies; ease of use, ease of manufacturing, ease of maintenance. It requires a total re-evaluation of the company, a restructuring, and a commitment to these principles. And it also requires working much more closely with customers, for it is from them that one gets the essential feedback to assess how well the process is going.

Putting Together the Product Team

There are numerous ways to form a product team, none of which is necessarily "the one best way." Different types of products, different industries, and even different corporate cultures might dictate differing styles of work. A brand new product, especially a potentially disruptive one, requires a very different process than a sustaining product, one that continues a long existing product line through incremental improvements and changes.

Remember the story of how the United States Navy does its work in Chapter 7 *(Being Analog)*? The Navy has two different organizational structures at work at the same time. One was a rigid, formal organization that handled the general assignment of personnel, their ranks, promotions, and administrative requirements. This is the formal rank structure of the Navy. But then, in the conduct of the daily work, the organizational structure was very different. Now rank was not so important, Instead, people formed functional teams according to their skills and levels of expertise, working together as a group, keeping track of the activity of the entire group, listening, critiquing, discussing, and learning.

This is an excellent model to follow. Mind you, the exactly organizational structure used by the Navy is probably not

directly applicable to other organizations, but it is the sprit that matters, the realization that there is no need to have a single fixed structure that applies to all activities. What is particularly nice about the Navy's structure is the flexibility, yet with a concentrated focus upon the task requirements. This is the spirit to emulate.

In design, the goal is that the actual development team be small, but that representatives from the larger community offer expert advice and be available when needed. Just how this is accomplished varies from situation to situation. This is the role of an informed management, and it is a difficult job.

User experience, UE, is itself at least six different areas of expertise (Chapter 9). Some people are skilled at several of these areas, but it is the rare individual who is skilled at all. As a result, this community requires several representatives. The several different disciplines that comprise UE often come from different academic backgrounds, ranging from the hard-core experimental science of psychology, to the art or architectural school background of the visual and industrial designers, to the literary skills and humanities background of the writers, to mechanical engineering expertise and programming skills required for rapid prototyping. So even within this single discipline, the people have different backgrounds, mean different things even when they use the same words, and come from very different cultures. Now mesh this mix with the technical and marketing people and you can see why working on a development team can be a real challenge: exciting during periods of great creativity and productivity and very frustrating during periods of non-communication, clashing personalities, and personal and technical disagreements.

The problems of creating products for people are simply an example of real life, with its diversity, synergies, challenges and clashes. Successful product teams, just as productive communities, learn to exploit their differences to create products that deliver useful value, are a pleasure to look at and to use, and that enhance people's lives.

The Organizational Structure of a Company

The traditional organizational structure of a company is that of a hierarchy. The company is divided into divisions, all reporting to the chief executive of the company, each reflecting either the product structure or their function: marketing, sales, manufacturing. Inside each division, the company is further divided into structures that reflect, as before, either different products or subcomponents of products, or function. Eventually, at the bottom, are those who do the work.

Organizational structures have many virtues, but the major one is accountability. Everyone knows to whom they report, and orders are expected to flow from the top downward, with information and feedback coming from the bottom upward. This organizational structure optimizes flow vertically up and down the organization: it makes it difficult for horizontal communication to take place, for a horizontal path cuts across departments and divisions. As a result, the traditional organizational structure tends to pit one part of the company against the other: it does not facilitate harmonious interaction and cooperation. In large companies, different divisions are often located at considerable geographical separation from one another, making cooperation even more difficult.

Products are usually produced horizontally across the structure. After all, to create a product requires the coordination and combination of many low-level people, from conception to shipping. This means that if the hierarchy of the company is drawn in a diagram from top to bottom, products are created and manufactured by the different groups structured horizontally across the bottom of the chart: See Figure 10.1.

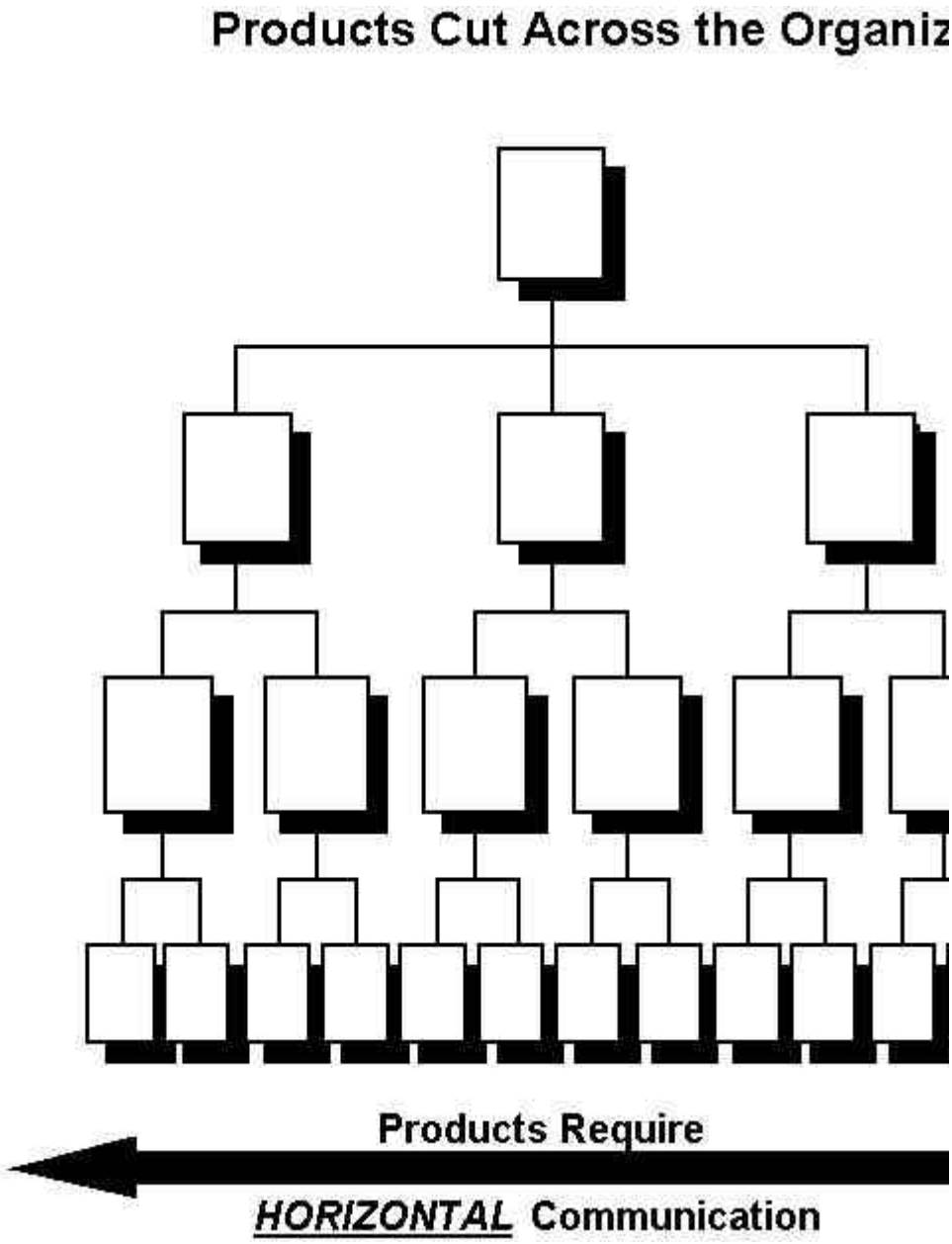

Figure 10.1 Traditional organization of a company is optimized for vertical communication. The traditional hierarchical structure of the company is not optimized for horizontal, cross-divisional interaction. Workers report to managers. Managers report to middle management, who report to the company executives. Information flows relatively smoothly up and down the organization. This organizational structure is idealized for command, for control, for accountability. It also provides clear reward structures and promotion paths. Hence, the traditional structure tends to pit one division against another. Moreover, the structure impedes horizontal communication, so it is less than optimal for the development of products.

There is a fundamental conflict between the traditional vertical organization of most companies and the functional, horizontal structure required to deliver products. Not only is the vertical organizational structure inimical to product development, but it leaves no place for functions that cut across the structure, functions such as usability, reliability, maintainability. It also makes design iteration difficult, because the organizational structure often leads to what has been called the “linear” or “waterfall” process of development: each stage of the process passes on its results to the stage afterwards, and once the pass-off has been made (once you are over the edge of the waterfall), there is no turning back.

The problem is that design is really an iterative process. Customers never know what they want until they have had a chance to use it, yet how can you build it in the first place for them to use if they don't know what they want? Answer: build it many times.

Iterative development means rapid prototyping. Work with the intended users, get an idea of their needs, do a quick

mockup in hours or days and try it out. The tryouts can be done in many ways. The prototype can be quick diagrams sketched on cardboard and foam, and the customers asked to imagine it as the final product, to imagine using it in their actual place of work.

In similar ways manufacturing engineers test the initial designs to understand how difficult it would be to produce the item. Maintenance engineers try to maintain them. And so on. The development process becomes a cooperative, iterative one. All participants cooperate. The initial stages of this development process often take longer and lead to more apparent frustration and complexity than the more common process, but the overall result is usually faster, less expensive, and superior.

This type of development is not a new idea. It is frequently written about, frequently practiced under many names: concurrent engineering, total quality control, virtual, activity centered design, and so on.

This style of iterative, concurrent development isn't easy to do. To many people, it appears ponderous: it starts off with meetings and the collection of user data rather than with "real work," It requires people with very different backgrounds, skills, and points of view to work together smoothly and harmoniously. It messes up the traditional lines of authority of a company. At first, progress seems very slow: there are few tangible, visible results of all those people, all those meetings. In the end, it is almost always faster and more efficient. Problems are addressed in the beginning, while fixes are still inexpensive. It is easy to change a design when it is still in prototype form, still done mainly with paper and foam or story boards. It is slow and expensive to change the design once it has been committed to prediction. Iterative, concurrent design it is definitely the correct way to proceed, but it requires changing the normal pattern of doing work. It requires changing the culture of the company. It may require reorganizing the company.

When the Reward Structure of a Company Gets in the Way

When it comes to working together, quite often two different companies find it easier to form a smooth, harmonious relationship than two different divisions of the same company. Why? Because of the reward structure.

In a company, individuals are rewarded and promoted based upon their performance. Higher-level employees might also receive part of their rewards based upon the performance of the business unit in which they work. All this sounds perfectly reasonable. In fact, it is specifically designed to get the best work out of each employee and to ensure that the employees work for the benefit of the business: If the business unit does well, the employees do well.

This reward structure has unintended consequences. The problem comes when a company has multiple business units, yet limited funds for bonuses or salary increases and limited positions available for promotion. The result is that employees are competing with one another for rewards rather than with competing companies.

Employees seek the path that is in their best interests. This is hardly a surprise, and in most businesses, that is what the reward structure is all about: it tries to arrange things so that the person's and the company's best interests are identical. They aren't. Oftentimes, an employee's most feared competitor is the person in another division of the same company, for the two are competing for the same promotion, salary raise, or bonus. It is in each person's best interest to do whatever it takes to improve the performance of their particular business unit, whether or not this is helpful to the competing business unit, even when this is not ideal for the company.

Suppose that during the course of designing a product, the development group realizes that a redesign would simplify manufacturing and maintenance, cutting the overall costs of the company and improving customer satisfaction. Do they dare do the redesign? Of course not. The redesign would not only delay the completion of their assigned tasks, but it is likely to cause them to exceed their budget. When it came time for the performance review of the people in the group, they would suffer because they had gone over budget and been late with the product. No matter that as a result, the company saved money. The maintenance and manufacturing engineers, who might not even have been aware of the valiant efforts of the development team, would be promoted and rewarded because they had lower costs.

The only way out of this dilemma is to ensure total accountability: rewards go to everyone in the process. The company has to be structured so that everyone is in it together. But this is not easy, not in a large company.

The reward structure has to be designed to take all critical variables into account. Thus, the charges for help lines, service and maintenance ought to go to the development team. Perhaps service should report to developers, the better to ensure that customer complaints are tallied and understood by the developers, the better to take account of them in the next product release. So too with the costs of these after-sales ventures. Out of sight, out of mind, goes the folk saying, and when the costs of help desks, service, and maintenance are out of sight, they are apt to play little role in the minds of the development team.

It's critically important to factor all costs of a product into the reward structure of the team, for otherwise, those aspects will be slighted in favor of the features that do receive awards. In a similar way, it is important that all benefits be credited

to the proper teams: product awards, increased sales, decreased service and maintenance, increased profit margins: all ought to be factored back to all who were involved.

Putting together a proper reward structure is never easy, but improper ones foster behavior that is not in the best interests of the company. The reward structure is perhaps the most important variable of all, more important that organizational structure. Put in the proper rewards and the appropriate behavior will follow.

Who Owns the Product?

"Marketing wants us to redesign the panel on the front of the product so that the features are more apparent in the store," a user experience designer complained to me, "but our tests shows that it also makes it harder to use. Now what do we do?"

This is a typical product tradeoff: in this case, ease of use versus sales appeal. Which point is correct? Both are legitimate points. Both are correct. But rather than conflict, what is needed is an organization that works like a team, where everyone realizes that the goal is products that sell better, that work better, and that give pride of ownership to the users. Products that are hard to use do not provide joy to their owners. But products that don't sell in the first place don't even give ease of use a chance. How does one make this tradeoff?

Note that similar issues will arise at all stages of the product process. Engineering wants a new feature, but it will add time to the schedule, perhaps an extra cost to the product. Marketing wants some new features. The User Experience team wants to hide the little-used controls behind a panel to make the device look less intimidating and confusing, but marketing shows that customers purchase products because of all the features, even if they never use them once they get home. Each of these issues pits one part of the development team against the others. Each has proponents who think they own that part of the product, that they should be making the decision.

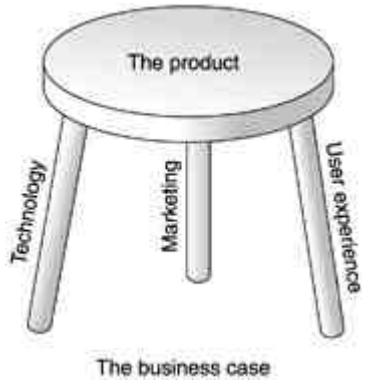

Figure 10.2: Who owns the product? The business. The foundation of the human-centered product development team is the business case for the product, and it is upon this foundation where ownership of the product resides. Disputes among the development team almost always result from the tensions inherent in product design: design is a series of tradeoffs where each aspect adds benefits and exhibits costs. Conflicts are inevitable. Moreover, each party is correct, only from their limited point of view. In the end, these tradeoffs must be decided upon with reference to the business casefor the product.

Who does own the development process? Who should make these decisions? The business. If there is anyone in charge of the product development process, it is those involved with the business side of the company: often the product manager. The human-centered development team itself has to consist of a team of equals, each contributing their own area of expertise. When there are conflicts, they must be resolved in an intelligent way, not by a power struggle. In most cases all parties are correct from their own point of view. The resolution requires taking a higher-level look at the problem and asking what decision is best for the business. What are the tradeoffs? How do they affect sales, customer satisfaction, the brand reputation and image, service, and maintenance? The decision is about business, not design. It has to be decided by the senior managers, those who hold the profit and loss accountability for the product.

Will a change increase functionality at a cost in schedule or price? It is resolved by a business analysis: how much is the market willing to pay? What are competitors doing? How important is that feature, both for the customer's needs and in terms of differentiating the product from those of customers? Will the addition or deletion of the feature make a noticeable impact on sales? If so, can a reasonable estimate be made of the impact? Is that impact worth the investment?

Is there a tradeoff in usability versus perceived functions at the point of sale? Can one estimate the increased sales that

would result from an emphasis on point of sales visibility? Can one estimate the impact on usability at the time of use: will it lead to increased service calls? Will it make a significant difference in customer satisfaction that would impact word-of-mouth recommendations and future purchases by this customer?

In other words, business issues get resolved by trying to assess the overall business impact of each alternative and then using those data guide the decisions. Sometimes, laying the matter out in this way helps find a compromise solution. Let the extra controls be hidden by a sliding panel that is kept open when the product ships, kept open on the showroom floor. Perhaps it isn't even attached, but is left for the user to install, the better to ensure that the controls are visible at point of sale. Sure, this adds a small burden on the purchaser, but it also acts as an effective compromise of the two legitimate competing interests.

Who owns the process? Nobody and everybody. The goal is better products for the consumer, better sales for the company, moreover, sales with less requirements for service and assistance. The goal is customers so satisfied and happy with the product that they become customers for life, buying more products, recommending the product, the company, and their experience to their friends.

Who owns the product? Everyone. Who makes the decisions when there are conflicts? Management. That's what management is for: to resolve conflicts, to agree that all sides have valid points, and then to use their judgment to make business decisions. The end result is always a tradeoff. Engineering emphasizes different aspects of the product than does marketing, which in turn has different emphases from user experience. They are all valid, they are all important. The development process is one of continual tradeoffs: marketability, functionality, usability. That's life in the world of business.

Corporate Requirements for Human-Centered Development

What does it take to do human-centered development in a company? Total commitment. Just as quality control required the company to have a total commitment, human-centered development has the same requirement. Here is a list of the fundamental requirements:

- **Total Corporate Commitment:** From lowest level worker to highest level management.
- **Organizational changes:** So designers and the eventual users of the product interact.
- **A formal, customer-centered product process:** The formal process puts the organizational stamp of approval on the process whereby technologists, marketing, and user-experience specialists work together in a team, as equal colleagues, from the very inception of the product through the shipping and assessment of market reaction. The process should propose and be built around an iterative design and study process. And finally, it should extend beyond the final release date of the product in order to collect filed data and user feedback on sales performance, repair and service requirements, usability and functionality that will drive the next release. This is where it is essential to listen to customers.
- **An engineering discipline of human-centered development:** So that when user-experience personnel are given the opportunity to work at all stages of the process, they are able to do so effectively, demonstrating their potential. This is especially difficult because the several disciplines that are involved have very diverse backgrounds which makes communication and a shared vision difficult. Nonetheless, a shared vision is essential.

Without the proper organizational structure, without the proper development process, even organizations with committed, dedicated individuals can still fail.

What Is the Proper Organizational Home for User Experience?

One standard question about the appropriate organization of a User Experience (UE) development group arises frequently: should there be one central group, with members farmed out to the various projects of the company or should the group be distributed, people hired to work wherever they are needed? Answer: it depends upon the company and its culture.

The development teams have to be a close-knit, working group. All the key development talent have to be together, working in concert. This is the point behind concurrent development. Do they have to belong to the same organization? No.

Having the development talent dispersed throughout the organization so that each individual works within and is assigned to the product group is my preferred way to get the work done. It's not the only possibility.

Having a central group has its value as well. Here, the group can have sufficient critical mass that they have all the skills required for UE development. Recall that I listed six critical skills for UE, a set unlikely to be found in a single individual. If UE is dispersed throughout the product groups, any individual group might not be able to afford the necessary skill sets. In addition, not all skills are required at all phases of the project, so with dispersed membership, there are times with nothing to do, times when the workers are overloaded. In a central group, there can be sufficient distribution over jobs to use all members effectively. And a central group allows for sharing of ideas and knowledge, so that the whole group continually grows in ability.

The central group has the difficulty of being removed from the action, being distant from the actual development activities. If there is a central group, then it is best to have it structured so that its members are assigned to particular projects, ideally moving to join those project teams.

There are other considerations as well. The social science and artistic talents of designers are not apt to be well understood by engineering or marketing organizations. When it comes to evaluations and promotions, the UE members can be at a disadvantage. Everyone can agree that they produce worthwhile results, but their skills are not apt to be rated as highly as that of the skilled, engineering specialist. For this purpose it is often better to be judged by peers who truly understand the contributions, the amount of effort required, and the kind of knowledge and talent involved.

I favor matrixed or virtual organizations. Let the UE team be hired and work for individual projects, but let there be a virtual central organization, with regular staff meetings and development seminars. Ensure that the leader of the virtual team is involved in project reviews as well as in personnel assessment. The virtual team can also serve as a central clearinghouse of talent, so that if a UE talent is needed for a project, perhaps that person could be borrowed from another project.

My impression is that organizations early in their understanding of the role of UE need a strong central group that reports high up in the hierarchy of the corporation, thereby giving the group visibility, authority and clout, with the members assigned to virtual or matrixed project teams. As the organization matures and begins to develop its understanding and appreciation for these skills, the need for a central team diminishes. In the end, it is probably not needed at all.

Many industrial design groups have always opted for a strong, central presence. Industrial and graphics design helps set the corporate identity, and the strong industrial design manager has always thought it important to have high visibility in the corporation, in many cases reporting directly to the CEO of the company. The reasoning is sound: this projects a strong message to everyone else that this is an area that is being taken very seriously by the company. It's nice if you can get it, but it isn't realistic, however, to expect UE to report at this level. Moreover, reporting to the corporate level of an organization can also be the kiss of death. It may isolate the group from the product teams where the work is being done. It tends to lead toward a police mentality — "we are the design police." Effective work is cooperative work. A complete disassociation from the groups with whom eventual cooperation is required is not a good thing.

The problems of effective management are complex. Any solution seems to have as many weaknesses as strengths. The real problem is our biological heritage: we work well in small groups, groups of less than ten, probably around five. Add to this the biological/cultural drives for status and recognition and there are the fundamental bases for conflicts. Small companies can often work smoothly and harmoniously. But when a company has thousands of employees — and some companies have hundreds of thousands, while governments and armies can have millions — then there is no correct answer: Whatever organizational structure is used will be difficult, complex, and never right.

The most important thing to remember is that it is results that matter. No organizational structure is perfect: every one has its problems. In the end, what matters are results.

Human-Centered Development Is Not Enough

I'm obviously a strong advocate of user experience in product development, a strong advocate of human-centered development. But let me emphasize, this is not enough.

"You keep saying ease-of-use is important," I am often admonished, "yet Apple had superior ease of use and it failed to become the market leader. How can you justify the emphasis on ease-of-use when you look at Apple's current position in the marketplace?" Good question. The story of Apple Computer is instructive.

The Macintosh initially did succeed. It had all three legs of the tripod in place: good technology, good marketing, and an excellent user experience, from industrial design through innovative ease of use. But, then, why did Apple as a company get into such difficulty? The answer is that although ease of use and high quality industrial design are important in the consumer market, it isn't enough. Human-centered development alone cannot save a product.

Note that two earlier products, The Xerox Star and the Apple Lisa had great design and ease of use, but they failed. They came before their time: the technology wasn't yet ready. As a result, they were too expensive, too slow, too limited in capability. They were easy to use, but there wasn't anything to do.

Apple Macintosh overcame these problems. Not only was it an elegant, human-centered development, but it came equipped with a set of useful programming applications, a printer that produced a pretty close reproduction of what was visible on the screen, and a price that, although high, was within the range of millions of customers. The Macintosh succeeded for almost ten years. That's a long time in this business.

In the long run, Apple failed because it chose a flawed marketing strategy. Oh yes, the tactical execution was superb, so Apple enjoyed an effective campaign, elegant advertisements, and a quick capturing of brand recognition. But the strategy achieved short-term gains at the expense of long-term viability.

Apple positioned itself as the maverick, the computer "for the rest of us." That is, Apple wanted nothing to do with traditional business, with the world of official forms and procedures, of the rigid working day with its rules and regulations. Apple was for the free-thinkers of society: artists, writers, students, youth. Bad idea. Because Apple shunned the world of business while IBM courted it, the IBM PC and its spreadsheet, Lotus 1-2-3, soon dominated that marketplace. The name "IBM" made the small, personal computer legitimate in the eyes of business in a way that the Apple II could never do.

Note that for any comparison of the products, Macintosh was far superior. The IBM PC was limited in power, complex and difficult to use. Its operating system, DOS, was rudimentary, and quirky. The Apple Macintosh was elegant, powerful and easy to use. It had a modern operating system: it had a graphical user interface, with windows, keyboard, and mouse and it could display and print graphics and a variety of fonts and styles. DOS had just the keyboard and commands that had to be memorized or, more often than not, looked up in a book. Printing graphics was a chore, as was displaying them on the screen. The DOS system had lots of manuals, and they had to be used: with the Macintosh, you never had to use the manual.

So what — DOS and the IBM PC was good enough to serve the needs of the business community. Apple restricted itself to the home, school, and the graphics arts industry. For a while, these were large enough to sustain the company, but by shunning business, Apple, had taken itself out of the market over the long run.

The marketplace positioning of Apple versus IBM was one of the factors in Apple's fall from grace. The Apple II was the first successful personal computer, so it initially had the benefits of being first. But IBM's courtship of industry led to its dominance in the business world. Not only did Apple not care, it bragged about the fact that it was not used in business.

Remember the argument about substitutable and nonsubstitutable goods? The market of non-substitutable goods leads to a winner-take-all business situation. Operating systems are nonsubstitutable goods. At first, the difference in infrastructures didn't matter, because the IBM PC dominated in business market whereas the Apple II dominated within the home and educational markets. However, in the decades that followed, these different markets tended to merge. People wanted the same computers at home as they used at work so they could bring their work home. Many schools felt it was their responsibility to use whatever the dominant system was for industry, whether or not this was appropriate for education. Because it is difficult for a company, school or household to cope with incompatible, conflicting operating systems, this tended to push the market toward one system, not on the basis of product superiority, but rather on the basis of convenience. Hence the domination of Microsoft/IBM DOS over the Apple II and later, the Macintosh operating system. The domination had its roots in those early days, and it has grown ever since. Today, decades later, the domination is almost complete, despite the dramatic changes in computer equipment since that time, despite that in those early days, the Apple Macintosh operating system was greatly superior to that of the IBM/Microsoft DOS.

When Apple introduced the Macintosh computer, it emphasized the virtues of user experience. The Macintosh was well developed from every point of view: superb technology, good industrial design, an excellent marketing campaign, and the best user experience in the business: it truly was easy to use — the average user never had to read the manuals. The Macintosh developed a loyal following. It did become the machine of choice for "the rest of us," for the non-business world, just as the advertising proclaimed. For a long time, Apple was able to grow rapidly and profitably by exploiting its

unrivaled core competency of "user experience." Apple charged a high price for its machines, but the customers were willing to make sacrifices to buy them: nothing else offered such usability and such emphasis on design, on graphical displays. Macintosh was especially popular in the school systems, among university students and faculty, and in the world of graphical arts, entertainment, and publishing. Emphasis on design paid off.

Business wants standards before elegance, and for historical reasons they had standardized on the IBM PC and, therefore, Microsoft DOS. Microsoft, meanwhile, understood the advantages of the Macintosh user experience, and therefore, after numerous efforts, it brought out a successful graphical user interface system of its own: Windows. By the fourth go-around, Microsoft had a successful product, Windows 95. Now, the advantage of the Macintosh was much diminished, and add to this the onslaught of an extensive marketing campaign, the desire of the computer users to have a single standard infrastructure, and soon the positive feedback cycle took hold. More companies developed software products for Windows than for Macintosh. As a result, even users who preferred their Macintosh felt compelled to switch to Windows in order to get the full range of products. And the more people that switched, the more other companies started making Windows products and stopped making Macintosh ones. Eventually the result is complete dominance of a market by a single standard.

Apple showed that superior user experience can make a difference, but that this alone isn't sufficient. Apple had technological superiority as well, but two legs of the tripod will not keep it standing: it needs all three legs. Apple's marketing leg emphasized the minority status of its customers. In fact its marketing slogan, "The computer for the rest of us," glorified the fact that it was not meant to be the solution for everyone: the slogan clearly implies that Apple intended to occupy a privileged minority position. A healthy minority to be sure, but a minority nonetheless. This is a bad idea when it comes to Operating Systems. When you have a nonsubstitutable good: you need to be the dominant player. This requires careful attention to the market, careful and continued leadership to ensure that yours is the most popular system. Apple failed to do this.

In the world of high-technology, any technical advantages are short-lived: others will soon catch up. The race is to the swift, but even the swift can lose if they don't always keep running. Remember Aesop's fables: this is how the tortoise beat the hare.

Nothing Worthwhile Is Easy

Remember Thomas Edison? The technology upon which he founded his numerous companies were usually first and best. He understood the importance of a systems analysis and product structure. He did everything right except for one thing: He failed to understand his customer. Many of Edison's companies failed.

In infrastructure products, marketshare is what matters. The market buys things based upon availability, functionality, price, and by the marketing images. The technology need only be "good enough."

Today, in the world of high-technology, we are victims of our own success. We have let technology lead the way, pushing ever faster to newer, faster, and more powerful systems, with nary a moment to rest, contemplate, and to reflect upon why, how, and for whom all this energy has been expended. In the process, although we have made many wonderful technological leaps, we have left humanity behind. Much of humanity is estranged from the technology, wary of it, fearful of it, and frightened of the future. These same emotions are increasingly common even among those of us who are technologically astute, who nominally are both friends and creators. Technology for the sake of technology is not an appropriate way to proceed.

There is a better way. It is possible to build systems that relate to people, that are task-centered, human-centered. But it requires high-technology businesses to change the way they do things, to move to simpler technologies that are a better match to people's activities. It requires a change to emphasize the human needs, to emphasize development for people. Such a change will not come easily. It requires a new process for product development, one that involves the social side of development as much as the engineering and marketing sides. It requires bringing a new discipline to the development table, user experience. And it requires that this new discipline live up to the challenges before it.

The current infrastructure for personal computers has outlived its usefulness. Although it was well designed for the problems and technology of the 1980s, it no longer works when faced with the problems and technologies of the 21st century. Moreover, the technology of the personal computer gets ever more complex every year. There is no way out, for it is caught in the tyranny of its own success, of its own marketing, and its business model that demands ever more features every year, thereby ever more complexity.

The only way out is through a disruptive technology, through a break with tradition to a technology that becomes invisible, that starts off focusing upon simplicity, joy of ownership, joy of use: a human-centered development philosophy.

The path to human-centered development, to simpler, better suited technology will not be easy. It requires changing some

long-honored ways of doing business. It requires restructuring the product process. It requires reorganizing the company. But the outcome is worthy. Along this path lies increased acceptance of technology, a technology that is invisible, thoroughly integrated into tools for the tasks and activities people wish to perform. The task is not easy because it involves people, organizations, and culture. Nothing worthwhile is easy.

> *The major problems facing the development of products that are safer, less prone to error, and easier to use and understand are not technological: they are social and organizational.*