cooper

www.cooper.com

4TH EDITION

ABOUT FACE

THE ESSENTIALS OF INTERACTION DESIGN

THE COMPLETELY UPDATED CLASSIC ON CREATING DELIGHTFUL USER EXPERIENCES

Alan Cooper, Robert Reimann, David Cronin, Chris Noessel

WILEY

CONTENTS

Part I: Goal-Directed Design
- CH 1: A Design Process for Digital Products
  - The Consequences of Poor Product Behavior
  - Why Digital Products Fail
  - Planning and Designing Product Behavior
  - Recognizing User Goals
  - Implementation Models and Mental Models
  - An Overview of Goal-Directed Design
- CH 2: Understanding the Problem: Design Research
  - Qualitative versus Quantitative Data in Design Research
  - Goal-Directed Design Research
  - Interviewing and Observing Users
  - Other Types of Qualitative Research
  - Research Is Critical to Good Design
- CH 3: Modeling Users: Personas and Goals
  - Why Model?
  - The Power of Personas
  - Why Personas Are Effective
  - Understanding Goals
  - Constructing Personas
  - Personas in Practice
  - Other Design Models
- CH 4: Setting the Vision: Scenarios and Design Requirements
  - Bridging the Research-Design Gap

- Scenarios: Narrative as a Design Tool
- Design Requirements: The “What” of Interaction
- The Requirements Definition Process

CH 5: Designing the Product: Framework and Refinement

- Creating the Design Framework
- Refining the Form and Behavior
- Validating and Testing the Design

CH 6: Creative Teamwork

- Small, Focused Teams
- Thinking Better, Together
- Working Across Design Disciplines
- The Extended Team
- Establishing a Creative Culture
- Identifying Skill Levels in Designers
- Collaboration Is the Key

Part II: Designing Behavior and Form

CH 7: A Basis for Good Product Behavior

- Design Values
- Interaction Design Principles
- Interaction Design Patterns

CH 8: Digital Etiquette

- Designing Considerate Products
- Designing Smart Products
- Designing Social Products

CH 9: Platform and Posture

- Product Platforms
- Product Postures

- Postures for the Desktop
- Postures for the Web
- Postures for Mobile Devices
- Postures for Other Platforms
- Give Your Apps Good Posture

CH 10: Optimizing for Intermediates
- Perpetual Intermediates
- Inflecting the Interface
- Designing for Three Levels of Experience

CH 11: Orchestration and Flow
- Flow and Transparency
- Orchestration
- Harmonious Interactions
- Motion, Timing, and Transitions
- The Ideal of Effortlessness

CH 12: Reducing Work and Eliminating Excise
- Goal-Directed Tasks versus Excise Tasks
- Types of Excise
- Excise Is Contextual
- Eliminating Excise
- Other Common Excise Traps

CH 13: Metaphors, Idioms, and Affordances
- Interface Paradigms
- Building Idioms
- Manual Affordances
- Direct Manipulation and Pliancy
- Escape the Grip of Metaphor

CH 14: Rethinking Data Entry, Storage, and Retrieval

- Rethinking Data Entry
- Rethinking Data Storage
- Rethinking Data Retrieval
- CH 15: Preventing Errors and Informing Decisions
  - Using Rich Modeless Feedback
  - Undo, Redo, and Reversible Histories
  - What If: Compare and Preview
- CH 16: Designing for Different Needs
  - Learnability and Help
  - Customizability
  - Localization and Globalization
  - Accessibility
- CH 17: Integrating Visual Design
  - Visual Art and Visual Design
  - The Elements of Visual Interface Design
  - Visual Interface Design Principles
  - Visual Information Design Principles
  - Consistency and Standards

Part III: Interaction Details

- CH 18: Designing for the Desktop
  - Anatomy of a Desktop App
  - Windows on the Desktop
  - Menus
  - Toolbars, Palettes, and Sidebars
  - Pointing, Selection, and Direct Manipulation
- CH 19: Designing for Mobile and Other Devices
  - Anatomy of a Mobile App
  - Mobile Navigation, Content, and Control Idioms

- Multi-Touch Gestures
- Inter-App Integration
- Other Devices

CH 20: Designing for the Web
- Page-Based Interactions
- The Mobile Web
- The Future

CH 21: Design Details: Controls and Dialogs
- Controls
- Dialogs
- Eliminating Errors, Alerts, and Confirmations
- The Devil Is in the Details

Appendix A: Design Principles
- Chapter 1
- Chapter 3
- Chapter 4
- Chapter 5
- Chapter 8
- Chapter 9
- Chapter 10
- Chapter 11
- Chapter 12
- Chapter 13
- Chapter 14
- Chapter 17
- Chapter 18
- Chapter 19
- Chapter 20

Chapter 21
Appendix B: Bibliography
Titlepage
Copyright
Credits
Dedication
About the Authors
Acknowledgments
Foreword
Introduction to the Fourth Edition
A Brief History of Interaction Design
IxD and User Experience
What This Book Is and What It Is Not
How This Book Is Structured
Changes Since the Third Edition
Examples Used in This Book
Who Should Read This Book
Advertisement
End-User License Agreement

List of Tables

List of Illustrations

Figure 1-1
Figure 1-2
Figure 1-3
Figure 1-4
Figure 1-5

Figure 1-6
Figure 1-7
Figure 1-8
Figure 2-1
Figure 2-2
Figure 3-1
Figure 3-2
Figure 3-3
Figure 3-4
Figure 3-5
Figure 3-6
Figure 3-7
Figure 4-1
Figure 5-1
Figure 5-2
Figure 5-3
Figure 5-4
Figure 5-5
Figure 6-1
Figure 6-2
Figure 6-3
Figure 6-4
Figure 6-5
Figure 7-1
Figure 7-2
Figure 9-1
Figure 9-2
Figure 9-3

Figure 9-4
Figure 9-5
Figure 9-6
Figure 9-7
Figure 9-8
Figure 9-9
Figure 9-10
Figure 9-11
Figure 9-12
Figure 9-13
Figure 9-14
Figure 9-15
Figure 10-1
Figure 10-2
Figure 11-1
Figure 11-2
Figure 11-3
Figure 11-4
Figure 11-5
Figure 11-6
Figure 11-7
Figure 11-8
Figure 11-9
Figure 11-10
Figure 12-1
Figure 12-2
Figure 12-3
Figure 12-4

Figure 12-5
Figure 12-6
Figure 12-7
Figure 12-8
Figure 12-9
Figure 12-10
Figure 12-11
Figure 12-12
Figure 12-13
Figure 12-14
Figure 12-15
Figure 12-16
Figure 12-17
Figure 12-18
Figure 13-1
Figure 13-2
Figure 13-3
Figure 13-4
Figure 13-5
Figure 13-6
Figure 13-7
Figure 13-8
Figure 13-9
Figure 13-10
Figure 14-1
Figure 14-2
Figure 14-3
Figure 14-4

Figure 14-5
Figure 14-6
Figure 14-7
Figure 14-8
Figure 14-9
Figure 15-1
Figure 15-2
Figure 15-3
Figure 15-4
Figure 15-5
Figure 15-6
Figure 15-7
Figure 16-1
Figure 16-2
Figure 16-3
Figure 16-4
Figure 16-5
Figure 16-6
Figure 16-7
Figure 16-8
Figure 16-9
Figure 17-1
Figure 17-2
Figure 17-3
Figure 17-4
Figure 17-5
Figure 17-6
Figure 17-7

Figure 17-8
Figure 18-1
Figure 18-2
Figure 18-3
Figure 18-4
Figure 18-5
Figure 18-6
Figure 18-7
Figure 18-8
Figure 18-9
Figure 18-10
Figure 18-11
Figure 18-12
Figure 18-13
Figure 18-14
Figure 18-15
Figure 18-16
Figure 18-17
Figure 18-18
Figure 18-19
Figure 18-20
Figure 18-21
Figure 18-22
Figure 18-23
Figure 18-24
Figure 18-25
Figure 19-1
Figure 19-2

[Figure 19-3]
[Figure 19-4]
[Figure 19-5]
[Figure 19-6]
[Figure 19-7]
[Figure 19-8]
[Figure 19-9]
[Figure 19-10]
[Figure 19-11]
[Figure 19-12]
[Figure 19-13]
[Figure 19-14]
[Figure 19-15]
[Figure 19-16]
[Figure 19-17]
[Figure 19-18]
[Figure 19-19]
[Figure 19-20]
[Figure 19-21]
[Figure 19-22]
[Figure 19-23]
[Figure 19-24]
[Figure 19-25]
[Figure 19-26]
[Figure 19-27]
[Figure 19-28]
[Figure 19-29]
[Figure 19-30]

Figure 19-31
Figure 19-32
Figure 19-33
Figure 19-34
Figure 19-35
Figure 19-36
Figure 19-37
Figure 19-38
Figure 19-39
Figure 19-40
Figure 19-41
Figure 20-1
Figure 20-3
Figure 20-4
Figure 20-5
Figure 20-6
Figure 20-7
Figure 20-8
Figure 20-9
Figure 20-10
Figure 20-13
Figure 21-1
Figure 21-2
Figure 21-3
Figure 21-4
Figure 21-5
Figure 21-6
Figure 21-7

Figure 21-8
Figure 21-9
Figure 21-10
Figure 21-11
Figure 21-12
Figure 21-13
Figure 21-14
Figure 21-15
Figure 21-16
Figure 21-17
Figure 21-18
Figure 21-19
Figure 21-20
Figure 21-21
Figure 21-22
Figure 21-23
Figure 21-24
Figure 21-25
Figure 21-26
Figure 21-27
Figure 21-28
Figure 21-29
Figure 21-30
Figure 21-31
Figure 21-32
Figure 21-33
Figure 21-34
Figure 21-35

Figure 21-36
Figure 21-37
Figure 21-38
Figure 21-39

Part I
Goal-Directed Design

- CH 1 A Design Process for Digital Products
- CH 2 Understanding the Problem: Design Research
- CH 3 Modeling Users: Personas and Goals
- CH 4 Setting the Vision: Scenarios and Requirements
- CH 5 Designing the Product: Framework and Design Refinement
- CH 6 Creative Teamwork

CH 1
A Design Process for Digital Products

This book has a simple premise: If we design and develop digital products in such a way that the people who use them can easily achieve their goals, they will be satisfied, effective, and happy. They will gladly pay for our products—and recommend that others do the same. Assuming that we can do so in a cost-effective manner, this will translate into business success.

On the surface, this premise seems obvious: Make people happy, and your products will be a success. Why, then, are so many digital products so difficult and unpleasant to use? Why aren't we all happy and successful when we use them? Why, despite the steady march of faster, cheaper, and more accessible technology, are we still so often frustrated?

The answer, in short, is the *absence of design* as a fundamental and equal part of the product planning and development process.

Design, according to industrial designer Victor Papanek, is *the conscious and intuitive effort to impose meaningful order*. We propose a somewhat more detailed definition of human-oriented design activities:

- Understanding the desires, needs, motivations, and contexts of people using products
- Understanding business, technical, and domain opportunities, requirements, and constraints
- Using this knowledge as a foundation for plans to create products whose form, content, and behavior are useful,

usable, and desirable, as well as economically viable and technically feasible

This definition is useful for many design disciplines, although the precise focus on form, content, and behavior varies depending on what is being designed. For example, an informational website may require particular attention to *content*, whereas the design of a simple TV remote control may be concerned primarily with *form*. As discussed in the Introduction, interactive digital products are uniquely imbued with complex *behavior*.

When performed using the appropriate methods, design can, and does, provide the missing human connection in technological products. But most current approaches to the design of digital products don’t work as advertised.

The Consequences of Poor Product Behavior

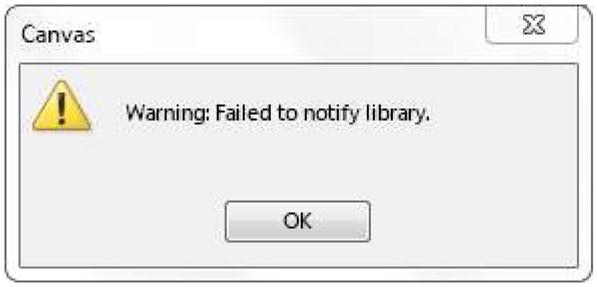
In the nearly 20 years since the publication of the first edition of *About Face*, software and interactive digital products have greatly improved. Many companies have begun to focus on serving people’s needs with their products and are spending the time and money needed to support the design process. However, many more still fail to do so—at their peril. As long as businesses continue to focus solely on *technology* and *market data* while shortchanging design, they will continue to create the kind of products we’ve all grown to despise.

The following sections describe a few of the consequences of creating products that lack appropriate design and thus ignore users’ needs and desires. How many of your digital products exhibit some of these characteristics?

Digital products are rude

Digital products often blame users for making mistakes that are not their fault, or should not be. Error messages like the one shown in Figure 1-1 pop up like weeds, announcing that the user has failed yet again. These messages also demand that the user acknowledge his failure by confirming it: OK.

Digital products and software frequently interrogate users, peppering them with a string of terse questions that they are neither inclined nor prepared to answer: “Where did you hide that file?” Patronizing questions like “Are you sure?” and “Did you really want to delete that file, or did you have some other reason for pressing the Delete key?” are equally irritating and demeaning.


Canvas

Warning: Failed to notify library.

Figure 1-1: Thanks for sharing. Why didn’t the application notify the library? Why did it want to notify the library? Why is it telling us? And what are we OKing, anyway? It is not OK that the application failed!

Our software-enabled products also fail to act with a basic level of decency. They forget information we tell them and don’t do a very good job of anticipating our needs. Even the iPhone—generally the baseline for good user experience on a digital device—doesn’t anticipate that someone might not want to be pestered with a random phone call when he is in the middle of a business meeting *that is sitting right there in the iPhone’s own calendar*. Why can’t it quietly put a call that isn’t from a family member into voicemail?

Digital products require people to think like computers

Digital products regularly assume that people are technology literate. For example, in Microsoft Word, if a user wants to rename a document she is editing, she must know that she must either close the document or use the "Save As…" menu command (and remember to delete the file with the old name). These behaviors are inconsistent with how a normal person thinks about renaming something; rather, they require that a person change her thinking to be more like the way a computer works.

Digital products are also often obscure, hiding meaning, intentions, and actions from users. Applications often express themselves in incomprehensible jargon that cannot be fathomed by normal users ("What is your SSID?") and are sometimes incomprehensible even to experts ("Please specify IRQ.").

Digital products have sloppy habits

If a 10-year-old boy behaved like some software apps or devices, he'd be sent to his room without supper. These products forget to shut the refrigerator door, leave their shoes in the middle of the floor, and can't remember what you told them only five minutes earlier. For example, if you save a Microsoft Word document, print it, and then try to close it, the application again asks you if you want to save it! Evidently the act of printing caused the application to think the document had changed, even though it did not. Sorry, Mom, I didn't hear you.

Software often requires us to step out of the main flow of tasks to perform functions that shouldn't require separate interfaces and extra navigation to access. Dangerous commands, however, are often presented right up front where users can accidentally trigger them. Dropbox, for example, sandwiches Delete between Download and Rename on its context menus, practically inviting people to lose the work they've uploaded to the cloud for safekeeping.

Furthermore, the appearance of software—especially business and technical applications—can be complex and confusing, making navigation and comprehension unnecessarily difficult.

Digital products require humans to do the heavy lifting

Computers and their silicon-enabled brethren are purported to be labor-saving devices. But every time we go out into the field to watch real people doing their jobs with the assistance of technology, we are struck by how much work they are forced to do simply to manage the proper operation of software. This work can be anything from manually copying (or, worse, retyping) values from one window into another, to attempting (often futilely) to paste data between applications that otherwise don't speak to each other, to the ubiquitous clicking and pushing and pulling of windows and widgets around the screen to access hidden functionality that people use every day to do their job.

The evidence is everywhere that digital products have a lot of explaining to do when it comes to their poor behavior.

Why Digital Products Fail

Most digital products emerge from the development process like a sci-fi monster emerging from a bubbling tank. Instead of planning and executing with a focus on satisfying the needs of the people who use their products, companies end up creating solutions that—while technically advanced—are difficult to use and control. Like mad scientists, they fail because they have not imbued their creations with sufficient humanity.

Why is this? What is it about the technology industry as a whole that makes it so inept at designing the interactive parts of digital products? What is so broken about the current process of creating software-enabled products that it results in such a mess?

There are four main reasons why this is the case:

- **Misplaced priorities** on the part of both product management and development teams
- **Ignorance about real users** of the product and what their baseline needs are for success
- **Conflicts of interest** when development teams are charged with both designing and building the user experience
- **Lack of a design process** that permits knowledge about user needs to be gathered, analyzed, and used to drive the development of the end experience

Misplaced priorities

Digital products come into the world subject to the push and pull of two often-opposing camps—marketers and developers. While marketers are adept at understanding and quantifying a marketplace opportunity, and at introducing and positioning a product within that market, their input into the product design process is often limited to lists of requirements. These requirements often have little to do with what users actually *need* or *desire* and have more to do with chasing the competition, managing IT resources with to-do lists, and making guesses based on market surveys—what people say they'll *buy*. (Contrary to what you might suspect, few users can clearly articulate their needs. When asked direct questions about the products they use, most tend to focus on low-level tasks or workarounds to product flaws. And, what they think they'll buy doesn't tell you much about how—or if—they will use it.)

Unfortunately, reducing an interactive product to a list of a hundred features doesn't lend itself to the kind of graceful orchestration that is required to make complex technology useful. Adding "easy to use" as a checklist item does nothing to improve the situation.

Developers, on the other hand, often have no shortage of input into the product's final form and behavior. Because they are in charge of construction, they decide exactly what gets built. And they too have a different set of imperatives than the product's eventual audience. Good developers are focused on solving challenging technical problems, following good engineering practices, and meeting deadlines. They often are given incomplete, myopic, confusing, and sometimes contradictory instructions and are forced to make significant decisions about the user experience with little time or knowledge of how people will actually use their creations.

Thus, the people who are most often responsible for creating our digital products rarely take into account the users' *goals*, needs, or motivations. At the same time, they tend to be highly reactive to market trends and technical constraints. This can't help but result in products that lack a coherent user experience. We'll soon see why goals are so important in addressing this issue.

The results of poor product vision are, unfortunately, digital products that irritate rather than please, reduce rather than increase productivity, and fail to meet user needs. Figure 1-2 shows the evolution of the development process and where, if at all, design has historically fit in. Most of digital product development is stuck in the first, second, or third step of this evolution, where design either plays no real role or becomes a surface-level patch on shoddy interactions—“lipstick on the pig,” as one of our clients called it. The core activities in the design process, as we will soon discuss, should *precede* coding and testing to ensure that products truly meet users’ needs.

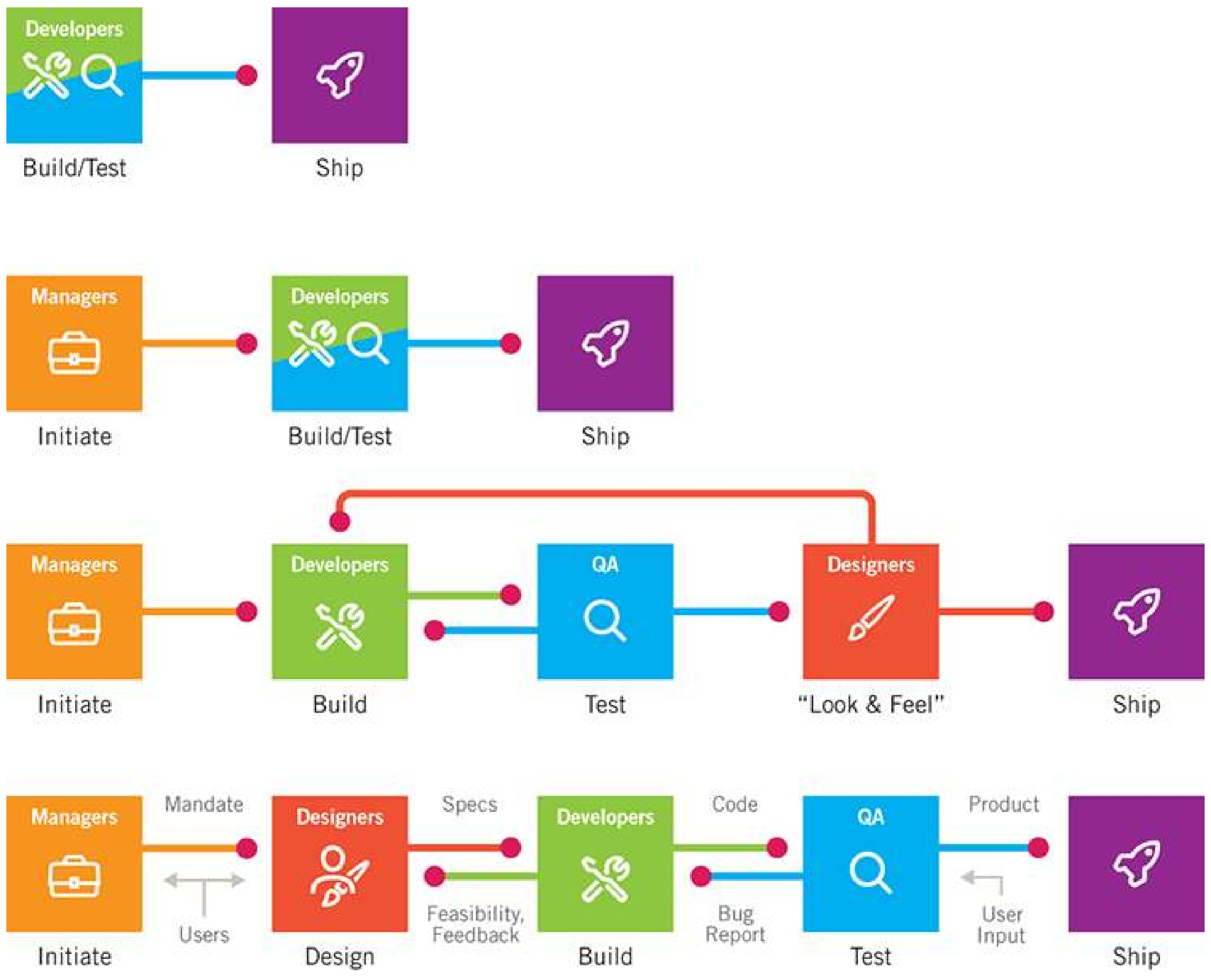


Figure 1-2: The evolution of the software development process. The first diagram depicts the early days of the software industry, when smart developers dreamed up products and then built and tested them. Inevitably, professional managers were brought in to help facilitate the process by translating market opportunities into product requirements. As depicted in the third diagram, the industry matured, and testing became a discipline in its own right. With the popularization of the graphical user interface (GUI), graphic designers were brought in to create icons and other visual elements. The final diagram shows the Goal-Directed approach to software development, where decisions about a product's capabilities, form, and behavior are made before the expensive and challenging construction phase.

Ignorance about real users

It's an unfortunate truth that the digital technology industry doesn't have a good understanding of what it takes to make users happy. In fact, most technology products get built without much understanding of users. We might know what *market segment* our users are in, how much money they make, how they like to spend their weekends, and what sorts of cars they buy. We might even have a vague idea of what kind of jobs they have and some of the major tasks they regularly perform. But does any of this tell us how to make them happy? Does it tell us *how* they will actually use the product we're building? Does it tell us *why* they are doing whatever it is they might need our product for, *why* they might want to choose our product over our competitors, or *how* we can make sure they do? No, it does not.

However, we should not give up hope. It is possible to understand our users well enough to make excellent products they will love. We'll see how to address the issue of understanding users and their behaviors with products in Chapters 2 and 3.

Conflicts of interest

A third problem affects the ability of vendors and manufacturers to make users happy. The world of digital product development has an important conflict of interest: The people who build the products—developers—are often also the people who design them. They are are also, quite understandably, the people who usually have the final say on what does and doesn't get built. Thus, developers often are required to choose between ease of coding and ease of use. Because developers' performance is typically judged by their ability to code efficiently and meet incredibly tight deadlines, it isn't difficult to figure out what direction most software-enabled products take. Just as we would never permit the prosecutor in a legal trial to also adjudicate the case, we should make sure that the people designing a product are not the same people building it. Even with appropriate skills and the best intentions, it simply isn't possible for a developer (or anyone, for that matter) to advocate effectively for the user, the business, and the technology all at the same time.

We'll see how to address the issue of building design teams and fitting them into the planning and development process in Chapter 6.

Lack of a design process

The last reason the digital product industry isn't cranking out successful, well-designed products is that it has no reliable *process* for doing so. Or, to be more accurate, it doesn't have a *complete* process for doing so. Engineering departments follow—or should follow—rigorous engineering methods that ensure the *feasibility* and quality of the technology. Similarly, marketing, sales, and other business units follow their own well-established methods for ensuring the commercial *viability* of new products. What's left out is a repeatable, predictable, and analytical process for ensuring **desirability**: *transforming an understanding of users into products that meet their professional, personal, and emotional needs*.

In the worst case, decisions about what a digital product will do and how it will communicate with users are simply a by-product of its construction. Developers, deep in their thoughts of algorithms and code, end up "designing" product behaviors in the same way that miners end up "designing" a landscape filled with cavernous pits and piles of rubble. In unenlightened development organizations, the digital product interaction design process alternates between the accidental and the nonexistent.

Sometimes organizations do adopt a design process, but it isn't quite up to the task. Many companies embrace the notion that integrating customers (or their theoretical proxies, domain experts) directly into the development process can solve human interface design problems. Although this has the salutary effect of sharing the responsibility for design with the user, it ignores a serious methodological flaw: confusing domain knowledge with design knowledge.

Customers, although they might be able to articulate the problems with an interaction, often cannot visualize the solutions to those problems. Design is a specialized skill, just like software development. Developers would never ask users to help them *code*; design problems should be treated no differently. In addition, customers who *purchase* a product may not be the same people who *use* it from day to day, a subtle but important distinction. Finally, experts in a domain may not be able to easily place themselves in the shoes of less-expert users when defining tasks and flows. Interestingly, the two professions that seem to most frequently confuse domain knowledge with design knowledge when building information systems—law and medicine—have notoriously difficult-to-use products. Coincidence? Probably not.

Of course, designers should indeed get feedback on their proposed solutions, both from users and the product team. But hearing about the problems is much more useful to designers—and better for the product—than taking proposed solutions from users at face value. In interpreting feedback, the following analogy is useful: Imagine a patient who visits his doctor with acute stomach pain. "Doctor," he says, "it *really* hurts. I think it's my appendix. You've got to take it out as soon as possible." A responsible physician wouldn't perform surgery based solely on a patient request, even an earnest one. The patient can describe the symptoms, but it takes the doctor’s professional knowledge to make the correct diagnosis and prescribe the treatment.

Planning and Designing Product Behavior

The planning of complex digital products, especially ones that interact directly with humans, requires a significant upfront effort by professional designers, just as the planning of complex physical structures that interact with humans requires a significant upfront effort by professional architects. In the case of architects, that planning involves understanding how the humans occupying the structure live and work, and designing spaces to support and facilitate those behaviors. In the case of digital products, the planning involves understanding how the humans using the product live and work, and designing product behavior and form that support and facilitate the human behaviors. Architecture is an old, well-established field. The design of product and system behavior—**interaction design**—is quite new, and only in recent years has it begun to come of age as a discipline. And this new design has fundamentally changed how products succeed in the marketplace.

In the early days of industrial manufacturing, engineering and marketing processes alone were sufficient to produce *desirable* products: It didn’t take much more than good engineering and reasonable pricing to produce a hammer, diesel engine, or tube of toothpaste that people would readily purchase. As time progressed, manufacturers of consumer products realized that they needed to differentiate their products from functionally identical products made by competitors, so design was introduced as a means to increase user desire for a product. Graphic designers were employed to create more effective packaging and advertising, and industrial designers were