

Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design

Larry L. Constantine & Lucy A. D. Lockwood

Review by Larry Wood

Obviously, this book makes a unique contribution to user interface design because of the authors' work in the development of essential use cases. More particularly, Constantine and Lockwood claim that trained usability specialists are in much too short a supply to fill demands for the foreseeable future. As a result, the authors argue "developers, the frontline troops of software, need to be empowered by better tools and techniques if they are to deliver highly usable software on restricted schedules and budgets. This book is about such methods." Frankly, I found the book to be quite informative and provocative from my perspective as a usability professional. I will leave it to "developers" out there to determine its utility for them.

The book is divided into six sections, titled: Toward More Usable Software, Essential Models for Usability, Creating the Visual Design, Completing the Design, Assessment and Improvement, and Organizing and Managing the Process

Chapter 1 begins with a description of the overly tedious process an airline ticket agent (not to mention the ticket holders) is subjected to when attempting to change a previously issued ticket. To quote the authors' description of an actual experience, "On several occasions, the agent would navigate her way to one screen, scroll through data, write down a code or identification number on a scrap of paper, and then navigate to another screen where she typed the same code again into another part of the system. Some 45 minutes and countless keyboard operations later, we had our new tickets and boarding passes and were

headed for the gate. It had taken us longer to check in at the airport than to drive there from 32 miles away." The remainder of chapter 1 concentrates on educating the reader about what effective usability is not. The authors systematically point out that while efforts to promote usability such as usability testing, development of style guides, rapid prototyping, and design reviews can be effective in improving it, they all have serious shortcomings.

In chapter 2 Constantine and Lockwood continue to lay the foundation for their approach to effective interface design, what they refer to Usage-Centered Design. They emphasize that it isn't sufficient just to focus on users, but more specifically, it is imperative to focus on their work and on providing usable tools for them. The authors then build a case for the importance of models as a basis for software design, pointing out the value of abstractions in hiding details and allowing designers to defer thinking about details of implementation and to maintain a focus on the larger picture.

Chapter 3 introduces five rules (intended to establish the general direction for software developers toward greater usability) and six principles (intended to provide more specific direction needed to resolve practical problems in user interface design). The five rules are: (1) access - by a knowledgeable user without help or instruction, (2) efficiency - for a skilled user, (3) progression - as a user gains experience, (4) support - of the real work of intended users, (5) context - suited to the actual conditions of use. The six principles of design are: (1) structure - obvious to

users, (2) simplicity, (3) visibility - of necessary options and materials, (4) feedback, (5) tolerance, and (6) reuse - maintain consistency.

Section 2 focuses on users, their work (represented as essential use cases), and the context in which the work takes place. The first part of chapter 4 is devoted to convincing the reader of the absolute importance of being involved with real end users, in addition to whatever information can be gleaned from sources such as managers, domain experts, and manuals. Having made such a strong case for working with real users, I found it rather odd that the details of how that should be done are left until the last section of the book. In the remainder of chapter 4, the authors introduce the notion of a user role and the process of creating user role models.

Once user roles are modeled, it is time to get down to understanding the work involved in a particular user role. That is the business of chapter 5, where the authors introduce the notions behind essential use cases. An essential use case is composed of a statement of the overall user purpose (or goal) plus a two-part narrative comprising the **user intention model** and the **system responsibility model**. The authors convincingly argue that essential use cases encourage the designers to focus on users' goals, which in turn, encourages them to consider a variety of concrete enactments (implementations to accomplish those goals).

In Chapter 6, the authors point out that a user interface must provide appropriate contexts and tools for enabling use cases. Providing those is what the

authors refer to as modeling the contents (or interaction contexts) of the interface and the relationships among the various contexts.

Section 3 reviews various issues and concerns behind the visual appearance and presentation of the interface. Chapter 7 discusses, very generally, several aspects of visual design (e.g., icons, with associated labels, fonts, use of color (and sound) and various aspects of screen organization. Chapter 8 discusses the choice and use of GUI widgets, including icons, menu organization and naming, and use of cascading and pop-up menus. The authors also discuss keyboard access, in keeping with their concern for efficiency, especially for more experienced users. Chapter 9 contains some suggestions about being usefully creative in complex situations for which standard widgets aren't quite up to the job. Several examples are provided using such techniques as visual persistence, animation, and overloading of controls.

In section 4 Constantine and Lockwood take a broad look at the processes and issues involved in turning use cases into a design, including the construction of prototypes, on-line help, accommodation of the progression from novice to experienced user, and special considerations of non-desktop interfaces, particularly the Web. Chapter 10 brings the essential use cases together with the visual design issues presented in the immediately preceding three chapters to discuss how they are realized in various prototypes (e.g., hi vs. lo fidelity, active vs. passive, vertical vs. horizontal). In chapter 11 there is a broad discussion of on-line help, where the authors suggest organizing on-line help by use cases. This is a relatively simple, but powerful concept, given that help systems are frequently organized in some non-usage-centered way (e.g., alphabetically or feature-oriented).

Chapter 12 addresses the problem of supporting evolving usage patterns. The authors make a strong case for the fact that most users are beginners only for a short while, and that the development of the GUI has tended to cater to

beginners, often at the expense of more experienced users. While not an easy set of problems to solve, the authors do an excellent job of articulating the competing needs of novice and expert users, while providing suggestions, guidelines, and some concrete suggestions for means to facilitate users' progression through intermediate and expert usage.

In chapter 13, the authors discuss, at some length, the factors surrounding the context of use. They revisit the issues of user roles regarding frequency of use and complexity of interaction and issues of the physical environment, such as temperature, noise, sources of information flow, and device constraints.

Chapter 14 is devoted to design issues in non-desktop software, emphasizing the Web, while including a smaller section devoted to embedded applications such as controls for VCRs and printers. In contrast to the popular claim that the Web is a new paradigm, the authors take the stance that the same usage-centered design rules and principles important in desktop applications apply equally well to creating usable Web sites. The chapter ends with an interesting discussion of usability issues that are unique to embedded applications and how they might be addressed.

Having previously discussed the development of essential use cases and how to turn those into a design, in chapter 15 the authors illustrate those concepts and processes with a case study. The case study centers on the development of a corporate telephone directory, and readers are encouraged to treat it as a tutorial of sorts. "Practice Breaks" are strategically placed, and readers are prompted to develop a solution to a particular step in the process before reading about the one provided by the authors. Here I think Constantine and Lockwood do a credible job of bringing together the various aspects of usage-centered design, as they have defined it.

Section 5 is devoted to assessment and improvement of the design. Chapter 16 discusses a variety of more or less formal assessment methods, including

expert evaluation, peer and user reviews, heuristic evaluation, and cognitive walk-throughs. In Chapter 17, the authors introduce a number of metrics for usability, falling into three broad categories: (user) preference metrics, performance metrics, and predictive metrics. Chapter 18 then provides an excellent introduction to usability testing, both in the laboratory and the field, including beta-testing. The authors include information on developing a test protocol, cautions against undue influence from those conducting the test, measuring performance, and in designing and planning the tests.

Section 6 is titled, "Organizing and Managing the Process", but as indicated earlier, the authors choose to delay discussing the details of working with users until this section. Prior to that topic, however, chapter 19 discusses issues surrounding the implementation of an interface design and is focussed on the merits of objected-oriented analysis, design.

While it seemed rather late in coming, chapter 20 provides a good introduction to the basics of working with users through interviews, observations and conducting a variety of meetings. The results are then used to produce what are referred to as "essential requirements", which are then used to create low fidelity designs (pencil and paper sketches) for refinement prior to any serious "coding." The authors also spend a considerable part of the chapter introducing their version of Joint Application Design, which they call Joint Essential Modeling.

In the final chapter, the Constantine and Lockwood discuss a variety of usability issues as they apply to the broader context of the development organization. These include various models of organization fit, organization usability standards and style guides, and the cultural issues surrounding the introduction and maintenance of a positive usability climate within an organization.

In summary, I found the book informative, generally well-written, and provocative because of its particular

perspective on development of usable software. In addition to the emphasis on essential use cases, some other items unique to this book are the authors' concern for support of evolving usage patterns, design of non-desk-top applications, and their tutorial approach to the case study in chap-

ter 15. The authors do cover a broad range of topics, which results in some not being developed in as much depth as some readers would prefer.

Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design by Larry L. Constantine and Lucy A. D. Lockwood

Publication date: 1999
ISBN: 0201924781
Published by Addison-Wesley Pub Co.

Larry Wood is SIGCHI Bulletin's book review editor. He can be reached at chi-bulletin-pubs@acm.org.

