

ALAN DIX, JANET FINLAY,
GREGORY D. ABOWD, RUSSELL BEALE

HUMAN-COMPUTER INTERACTION

THIRD EDITION



Maxell, Ltd.
Ex. 2010
Page 1 of 6

PEARSON
Prentice
Hall

HUMAN—COMPUTER INTERACTION

Third Edition

Alan Dix, *Lancaster University*

Janet Finlay, *Leeds Metropolitan University*

Gregory D. Abowd, *Georgia Institute of Technology*

Russell Beale, *University of Birmingham*



Harlow, England • London • New York • Boston • San Francisco • Toronto • Sydney • Singapore • Hong Kong
Tokyo • Seoul • Taipei • New Delhi • Cape Town • Madrid • Mexico City • Amsterdam • Munich • Paris • Milan

Maxell, Ltd.
Ex. 2010
2 of 6

Pearson Education Limited

Edinburgh Gate

Harlow

Essex CM20 2JE

England

and Associated Companies throughout the world

Visit us on the world wide web at:

www.pearsoned.co.uk

First published 1993

Second edition published 1998

Third edition published 2004

© Prentice-Hall Europe 1993, 1998

© Pearson Education Limited 2004

The rights of Alan Dix, Janet E. Finlay, Gregory D. Abowd and Russell Beale to be identified as authors of this work have been asserted by them in accordance with the Copyright, Designs and Patents Act 1988.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without either the prior written permission of the publisher or a licence permitting restricted copying in the United Kingdom issued by the Copyright Licensing Agency Ltd, 90 Tottenham Court Road, London W1T 4LP.

All trademarks used herein are the property of their respective owners. The use of any trademark in this text does not vest in the author or publisher any trademark ownership rights in such trademarks, nor does the use of such trademarks imply any affiliation with or endorsement of this book by such owners.

ISBN-13: 978-0-13-046109-4

ISBN-10: 0-13-046109-1

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library

10 9 8 7 6 5 4 3

10 09 08 07 06

Typeset in 10/12¹/₂pt Minion by 35

Printed and bound by Scotprint, Haddington

trade-off between appearance and ease of use may mean that this is the right thing to do, but you should take care before confusing the user needlessly.

Chic design is also a problem in physical devices. One of the authors was once in a really high-class hotel and found he could not locate the flush in the toilet. Only after much fumbling did he discover that one of the tiles could be pressed. The ‘active’ tile was level with the rest of the tiled wall – a very clean design, but not very usable!

You then need to know *where you are going* when you click a button or *what will happen*. Of course you can try clicking the button to see. In the case of a website or information system this may mean you then have to use some sort of ‘back’ mechanism to return, but that is all; however, in an application or device the action of clicking the button may already have caused some effect. If the system has an easy means to undo or reverse actions this is not so bad, but it is better if users do not have to use this ‘try it and see’ interaction. Where response times are slow this is particularly annoying.

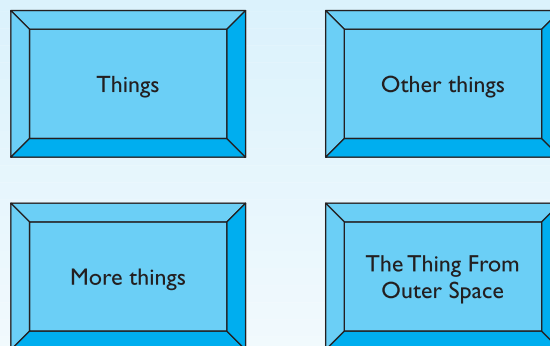
Remember too that icons are typically not self-explanatory and should always be accompanied by labels or at the very least tooltips or some similar technique. A picture paints a thousand words, but typically only when explained first using fifteen hundred!

DESIGN FOCUS



Beware the big button trap

Public information systems often have touchscreens and so have large buttons. Watch someone using one of these and see how often they go to the wrong screen and have to use ‘back’ or ‘home’ to try again. If you look more closely you will find that each button has only one or two words on it giving the title of the next screen, and possibly some sort of icon. Quite rightly, the button label will be in a large font as users may have poor eyesight.



It is hard to choose appropriate labels that mean the same for everyone, especially when the breadth of the screen hierarchy is fixed by the maximum number of buttons. So it is no wonder that people get confused. However, there is usually plenty of room for additional explanation in a smaller font, possibly just the next level of button labels, or a sentence of explanation. It may not look as pretty, but it may mean that people actually find the information they are looking for.

Special care has to be taken if the same command or button press means something different in different contexts. These different contexts that change the interpretation of commands are called *modes*. Many older text editors would interpret pressing 'x' to mean 'enter me into the text' in a normal typing mode, but 'exit' in a special command mode. If modes are clearly visible or audible this is less of a problem and in Chapter 3 (Section 3.6.7) we saw how palettes are one way to achieve this. In general, modes are less of a problem in windowed systems where the mode is made apparent by the current window (if you remember which it is). However, physical devices may have minimal displays and may be operated without visual attention.

Finally, if you have just done some major action you also want some sort of confirmation of *what you've done*. If you are faultless and have perfect knowledge, of course you will be sure that you have hit the right key and know exactly what

DESIGN FOCUS



Modes

Alan's mobile phone has a lock feature to prevent accidental use. To remove the lock he has to press the 'C' (cancel) button which then asks for an additional 'yes' to confirm removing the lock. So, in 'locked' mode, 'C' followed by 'yes' means 'turn off lock' and these are the most frequent actions when Alan takes the phone from his pocket.

However, Alan is forgetful and sometimes puts the phone in his pocket unlocked. This leads to occasional embarrassing phone calls and also to another problem.

The 'yes' button is quite big and so this is often pressed while in his pocket. This puts the phone into 'dial recent numbers' mode with a list of recent calls on screen. In this mode, pressing 'C' gives a prompt 'delete number' and pressing 'yes' then deletes the number from the phone's address book. Unhappily, this often means he takes the phone from his pocket, automatically presses 'C', 'yes' only to see as he looks down to the handset the fatal words 'number deleted'. Of course there is no undo!



will happen. Remember, too, that to know what will happen, you would need to know everything about the internal state of the system and things outside, like the contents of files, networked devices, etc., that could affect it. In other words, if you were omniscient you could do it. For lesser mortals the system needs to give some *feedback* to say what has happened.

In an information system, there is a related but slightly different issue, which is to know *where you have been*. This helps you to feel in control and understand your navigation of the information space. The feeling of disorientation when you do not have sufficient means to know where you are and where you have been has been called ‘lost in hyperspace’. Most web browsers offer a history system and also a ‘back’ button that keeps a list of recently visited pages.

5.6.2 Global structure – hierarchical organization

We will now look at the overall structure of an application. This is the way the various screens, pages or device states link to one another.

One way to organize a system is in some form of hierarchy. This is typically organized along functional boundaries (that is, different kinds of things), but may be organized by roles, user type, or some more esoteric breakdown such as modules in an educational system.

The hierarchy links screens, pages or states in logical groupings. For example, Figure 5.6 gives a high-level breakdown of some sort of messaging system. This sort of hierarchy can be used purely to help during design, but can also be used to structure the actual system. For example, this may reflect the menu structure of a PC application or the site structure on the web.

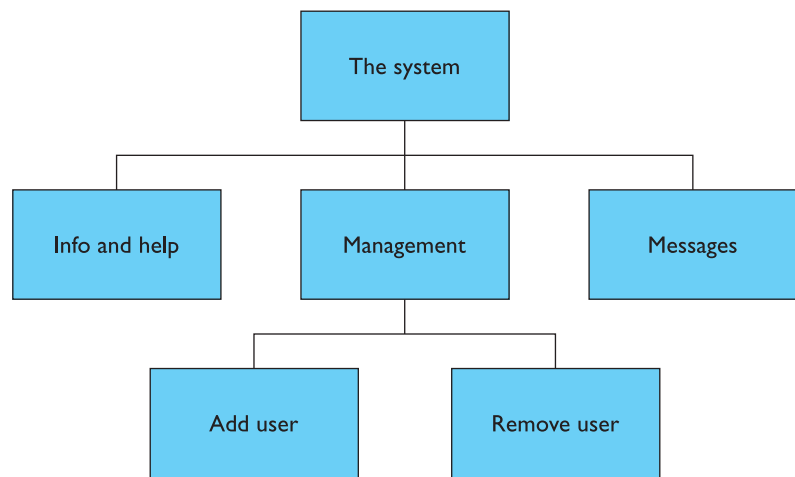


Figure 5.6 Application functional hierarchy