

Human Computer Interaction Development & Management



Tonya Barrier



IRM PRESS

Human Computer Interaction Development and Management

Tonya Barrier, Ph.D.
Southwest Missouri State University, USA



IRM Press

Publisher of innovative scholarly and professional
information technology titles in the cyberage

Hershey • London • Melbourne • Singapore • Beijing

Acquisitions Editor: Mehdi Khosrow-Pour
Managing Editor: Jan Travers
Assistant Managing Editor: Amanda Appicello
Copy Editor: Amanda Appicello
Cover Design: Tedi Wingard
Printed at: Integrated Book Technology

Published in the United States of America by
IRM Press
1331 E. Chocolate Avenue
Hershey PA 17033-1117
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@idea-group.com
Web site: <http://www.irm-press.com>

and in the United Kingdom by
IRM Press
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 3313
Web site: <http://www.eurospan.co.uk>

Copyright © 2002 by Idea Group Inc. All rights reserved. No part of this book may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher.

Library of Congress Cataloguing-in-Publication Data

Human computer interaction development and management / [edited by] Tonya Barrier.
p. cm.

Includes bibliographical references and index.

ISBN 1-931777-13-6 (paper)

1. Human-computer interaction. I. Barrier, Tonya, 1959-

QA76.9.H85 H8565 2002
004'.01'9--dc21

2002017315

eISBN: 1-931777-35-7

British Cataloguing-in-Publication Data

A Cataloguing-in-Publication record for this book is available from the British Library.



Other New Releases from IRM Press

- **Effective Healthcare Information Systems**, Adi Armoni (Ed.)
ISBN: 1-931777-01-2 / eISBN: 1-931777-20-9 / approx. 340 pages / US\$59.95 / © 2002
- **Data Warehousing and Web Engineering**, Shirley Becker (Ed.)
ISBN: 1-931777-02-0 / eISBN: 1-931777-21-7 / approx. 334 pages / US\$59.95 / © 2002
- **Information Technology Education in the New Millennium**, Mohammad Dadashzadeh, Al Saber and Sherry Saber (Eds.) /
ISBN: 1-931777-05-5 / eISBN: 1-931777-24-1 / approx. 308 pages / US\$59.95 / © 2002
- **Information Technology Management in Developing Countries**, Mohammad Dadashzadeh (Ed.) / ISBN: 1-931-777-03-9 / eISBN: 1-931777-23-3 / approx. 348 pages / US\$59.95 / © 2002
- **Strategies for eCommerce Success**, Bijan Fazlollahi (Ed.)
ISBN: 1-931777-08-7 / eISBN: 1-931777-29-2 / approx. 352 pages / US\$59.95 / © 2002
- **Collaborative Information Technologies**, Mehdi Khosrow-Pour (Ed.)
ISBN: 1-931777-14-4 / eISBN: 1-931777-25-X / approx. 308 pages / US\$59.95 / © 2002
- **Web-Based Instructional Learning**, Mehdi Khosrow-Pour (Ed.)
ISBN: 1-931777-04-7 / eISBN: 1-931777-22-5 / approx. 322 pages / US\$59.95 / © 2002
- **Modern Organizations in Virtual Communities**, Jerzy Kisielnicki (Ed.)
ISBN: 1-931777-16-0 / eISBN: 1-931777-36-5 / approx. 316 pages / US\$59.95 / © 2002
- **Enterprise Resource Planning Solutions and Management**, Fiona Fui-Hoon Nah (Ed.)
ISBN: 1-931777-06-3 / eISBN: 1-931777-26-8 / approx. 308 pages / US\$59.95 / © 2002
- **Interactive Multimedia Systems**, Syed M. Rahman (Ed.)
ISBN: 1-931777-07-1 / eISBN: 1-931777-28-4 / approx. 314 pages / US\$59.95 / © 2002
- **Ethical Issues of Information Systems**, Ali Salehnia (Ed.)
ISBN: 1-931777-15-2 / eISBN: 1-931777-27-6 / approx. 314 pages / US\$59.95 / © 2002
- **Intelligent Support Systems: Knowledge Management**, Vijay Sugumaran (Ed.)
ISBN: 1-931777-00-4 / eISBN: 1-931777-19-5 / approx. 318 pages / US\$59.95 / © 2002
- **Human Factors in Information Systems**, Edward Szewczak and Coral Snodgrass (Eds.)
ISBN: 1-931777-10-1 / eISBN: 1-931777-31-4 / approx. 342 pages / US\$59.95 / © 2002
- **Global Perspective of Information Technology Management**, Felix B. Tan (Ed.)
ISBN: 1-931777-11-4 / eISBN: 1-931777-32-2 / approx. 334 pages / US\$59.95 / © 2002
- **Successful Software Reengineering**, Sal Valenti (Ed.)
ISBN: 1-931777-12-8 / eISBN: 1-931777-33-0 / approx. 330 pages / US\$59.95 / © 2002
- **Information Systems Evaluation Management**, Wim van Grembergen (Ed.)
ISBN: 1-931777-18-7 / eISBN: 1-931777-37-3 / approx. 336 pages / US\$59.95 / © 2002
- **Optimal Information Modeling Techniques**, Kees van Slooten (Ed.)
ISBN: 1-931777-09-8 / eISBN: 1-931777-30-6 / approx. 306 pages / US\$59.95 / © 2002
- **Knowledge Mapping and Management**, Don White (Ed.)
ISBN: 1-931777-17-9 / eISBN: 1-931777-34-9 / approx. 348 pages / US\$59.95 / © 2002

Excellent additions to your institution's library!

Recommend these titles to your Librarian!

**To receive a copy of the IRM Press catalog, please contact
(toll free) 1/800-345-4332, fax 1/717-533-8661,
or visit the IRM Press Online Bookstore at: [<http://www.irm-press.com>]**

Note: All IRM Press books are also available as ebooks on netlibrary.com as well as other ebook sources. Contact Ms. Carrie Stull at [cstull@idea-group.com] to receive a complete list of sources where you can obtain ebook information or IRM Press titles.

Human Computer Interaction Development and Management

Table of Contents

Foreword	vii
Preface	viii
Chapter 1. Towards User-Oriented Control of End-User Computing in Large Organizations	1
<i>Neil McBride, DeMontfort University, United Kingdom</i>	
<i>A. Trevor Wood-Harper, University of Salford, United Kingdom and University of South Australia, Australia</i>	
Chapter 2. On-Line User Interaction with Electronic Catalogs: Language Preferences Among Global Users	18
<i>Aryya Gangopadhyay and Zhensen Huang</i>	
<i>University of Maryland Baltimore County, USA</i>	
Chapter 3. End Users as Expert System Developers?	31
<i>Christian Wagner, City University of Hong Kong, China</i>	
Chapter 4. Designing End-User Geographic Information Systems	53
<i>Lawrence West, Jr., University of Central Florida, USA</i>	
Chapter 5. Hypermedia Document Management: A Metadata and Meta-Information System	71
<i>Woojong Suh and Heeseok Lee</i>	
<i>Korea Advanced Institute of Science and Technology, Korea</i>	
Chapter 6. An Adaptive Probe-Based Technique to Optimize Join Queries in Distributed Internet Databases	93
<i>Latifur Khan, University of Texas at Dallas, USA</i>	
<i>Dennis McLeod and Cyrus Shahabi, University of Southern California, USA</i>	

Chapter 7. Strategies for Managing EUC on the Web	117
<i>R. Ryan Nelson, University of Virginia, USA</i>	
<i>Peter Todd, University of Houston, USA</i>	
Chapter 8. Exploring the Measurement of End User Computing Success	134
<i>Conrad Shayo, California State University of San Bernardino, USA</i>	
<i>Ruth Guthrie, California Polytechnic University of Pomona, USA</i>	
<i>Magid Igbaria, Claremont Graduate University, USA</i>	
Chapter 9. Constructive Design Environments: Implementing End-User Systems Development	153
<i>John G. Gammack, Murdoch University, Australia</i>	
Chapter 10. An Information Systems Design Framework for Facilitating TQM Implementation	174
<i>Nazim U. Ahmed, Ball State University, USA</i>	
<i>Ramarathnam Ravichandran, Design Systems, USA</i>	
Chapter 11. Methodology of Schema Integration for New Database Applications: A Practitioner's Approach	194
<i>Joseph Fong, City University of Hong Kong, China</i>	
<i>Kamalakar Karlapalem, Hong Kong University of Science & Technology, China</i>	
<i>Qing Li and Irene Kwan, Hong Kong Polytechnic University, China</i>	
Chapter 12. CMU-WEB: A Conceptual Model for Designing Usable Web Applications	219
<i>Akhilesh Bajaj and Ramayya Krishnan</i>	
<i>Carnegie Mellon University</i>	
Chapter 13. The Effects of Using a Triangulation Approach of Evaluation Methodologies to Examine the Usability of a University Website	243
<i>Dana H. Smith, Zhensen Huang, Jennifer Preece and Andrew Sears</i>	
<i>University of Maryland, Baltimore County, USA</i>	
Chapter 14. Adaptive Web Representation	255
<i>Arno Scharl, Vienna University of Economics, Austria</i>	

Chapter 15. Usability: Changes in the Field – A Look at the System Quality Aspect of Changing Usability Practices	261
<i>Leigh Ellen Potter, Griffith University, Queensland, Australia</i>	
Chapter 16. Facilitating End User Database Development by Working with Users' Natural Representations of Data	271
<i>Valerie J.Hobbs and Diarmuid J. Pigott Murdoch University, Australia</i>	
Chapter 17. User Developed Applications: Can End Users Assess Quality?	289
<i>Tanya J. McGill, Murdoch University, Australia</i>	
Chapter 18. Toward an Understanding of the Behavioral Intention to Use A Groupware Application	304
<i>Yining Chen and Hao Lou, Ohio University, USA</i>	
About the Editor	314
Index	315

Foreword

With the advent of new technology and new software, comes the management of the information systems of the organization. Technology and software development is at an all time high. Management of the information systems area is very complex and volatile.

Organizations today realize that information systems must be managed. Organizations cannot continue to blindly accept and introduce components into information systems without studying the effectiveness, feasibility and efficiency of the individual components of their information systems. Information systems may be the only business area where it is automatically assumed that the “latest, greatest and most powerful component is the one for our organization.” Information systems must be managed and developed as any other resource in organizations today.

The purpose of this book is to collect articles concerning the management and development of information systems so that organizations can effectively manage information systems growth and development in their organization.

The management of information systems within the organization is a diverse area. Not only must hardware, software, data, information, and networks must be managed, but also, people must be managed. Humans must be trained to use information systems. Systems must be developed so humans can use the systems as efficiently and effectively as possible. Therefore, topics included in this book concern human computer interaction such as training, aesthetics, ergonomics, and user friendliness. Questions posed may be: What monitor size is best? What desk height is best? Which colors should I use on outputs? What kinds of hardware should I provide for my physically challenged workers? How should I build a workstation to reduce problems such as carpal tunnel syndrome? What kinds of training programs are best? When should we update our hardware and software? The list of questions regarding the physical requirements for humans is infinite. However, the topic of human computer interaction is not complete without the study

of organizations, humans and information systems.

Organizations have changed with the introduction of technology. The Internet and extranet along with the concept of electronic messaging systems have changed the way organizations communicate. On the whole, organizations have increased and improved communications. However, these same communication channels have introduced more IT security and more problems especially with the “new e-mail” viruses. These concepts must be managed.

Organizational structural changes have been made because organizations expect individuals to be more productive as technology is introduced. Such organizations are continually “right sizing” and changing roles as technology changes. Today most individuals are responsible for many of their “own” technological needs. These concepts must be managed.

Employee training, and management training is evolving. The introduction of IT has produced new mediums for development and training such as online multimedia training to group decision making using IT. These concepts must be managed.

It would be impossible to list individually the topics concerning human computer interaction development, organizations, organizations changes, new technology and the management of IT. The purpose of this book is to gather a useful set of articles to describe human computer interaction development and management of organizations. The authors of the individual manuscripts have written the articles to further the effective management and development of IT in organizations. I invite you to peruse the book to find the article that best suits your needs.

Tonya Barrier
Southwest Missouri State University, USA

Preface

The human component of information systems use is an often overlooked, but extremely important factor. The end user has to deal with all the problems with the systems and understand how to utilize all the components of a given system in order for the entire operation to be optimized. Many organizations are looking to the end user when designing and implementing new systems, but in order to understand what role these end users can and should play, organization heads and project managers need to have access to the latest information regarding the human factor in information systems development and management. This timely new book provides the most up-to-date reporting on research and practice in the fields of end user computing and human computer interaction. From geographic information systems to online catalogs, the chapters in this book cover a wide range of topics related to end users and provide practical as well as theoretical guidance on how to best incorporate the human factor into design and management decisions. The authors from a wide variety of organizational and cultural backgrounds, and experts in their field share their insights in the following chapters.

Chapter 1 entitled, “Towards User-Oriented Control of End-User Computing in Large Organizations” by Neil McBride of DeMontfort University and A. Trevor Wood-Harper of the University of Salford (United Kingdom) and the University of South Australia concentrates an IT-oriented view with an alternative user-oriented view. The chapter advocates a shift in End User Computing research away from the technology and the IT issues towards the political, social and cultural issues associated with the users. The chapter proposes a dynamic model for EUC in which the progression of EUC within an organization is visualized as a series of inference loops.

Chapter 2 entitled, “Online User Interaction with Electronic Catalogs: Language Preferences Among Global Users” by Aryya Gangopadhyay and Zhensen Huang of the University of Maryland-Baltimore County (USA) describes a bilingual electronic catalog that can be used by online retailers for selling products and/or services to customers in either English or Chinese. The chapter reports on three separate usages of the catalog: browsing, direct search and exact matches. The authors test the efficiency of usage by measuring time spent as well as studying the path followed by the user in retrieving information in all of the above scenarios.

Chapter 3 entitled, “End Users as Expert Systems Developers?” by Christian Wagner of City University of Hong Kong (China) discusses the differences associated with end user development, both in terms of design quality and knowledge content. The chapter is based upon an analysis of 25 expert systems written by non-professional developers. The report of the analysis within the chapter reveals significant quality and size limitations that indicate limited feasibility of end user expert system development.

Chapter 4 entitled, “Designing End-User Geographic Information Systems” by Lawrence West, Jr. of the University of Central Florida (USA) identifies the concepts most needed for end user geographic information systems (GIS) use and suggests remedial efforts to reduce the burden of system operation and improve data integrity. The chapter presents useful guidelines and offers approaches, which make extensive use of metadata storage. These approaches may be implemented as tools in GIS software provided to end-users.

Chapter 5 entitled, “Hypermedia Document Management: A Metadata and Meta-Information System” by Woojong Suh and Heeseok Lee of Korea Advanced Institute of Science and Technology (Korea) identifies metadata roles and components necessary to build a metadata schema. The authors propose a meta-information system, Hyperdocument Meta-Information systems (HyDoMiS), that performs three functions, metadata management, search and reporting. The authors indicate that this system will help to implement and maintain hypermedia information systems effectively.

Chapter 6 entitled, “An Adaptive Probe-based Technique to Optimize Join Queries in Distributed Internet Databases” by Latifur Khan of the University of Texas at Dallas, Dennis McLeod and Cyrus Shahabi of the University of Southern California (USA) discusses an experiment that consisted of two servers running the same DBMS connected to the Internet. The authors discuss how a static query optimizer could choose an expensive plan by mistake due to its lack of knowledge about the run time environment, inaccurate statistical assumptions in size estimation or neglect of the cost of remote method invocation. The authors present a probing mechanism with an adaptive technique that offers a more cost effective approach than the static query optimizer.

Chapter 7 entitled, “Strategies for Managing EUC on the Web” by R. Ryan Nelson of the University of Virginia and Peter Todd of the University of Houston (USA) examines which strategies organizations are using to maximize the benefits of the Web for end users while mitigating the inherent risks. The authors surveyed individuals from 12 organizations and report the results of their survey in this chapter. The results indicate that organizations are doing an adequate job of establishing roles, standards, and mechanisms; however, their efforts for resource allocations, development management and maintenance are lacking.

Chapter 8 entitled, “Exploring the Measurement of End User Computing Success” by Conrad Shayo of California State University of San Bernardino, Ruth Guthrie of California Polytechnic University of Pomona and Magid Igbaria of Claremont Graduate School (USA) explores the literature on EUC success measurement and discusses the main issues and concerns researchers face. The authors offer recommendations to optimize success measurement including using unobtrusive measures of success, taking into account contextual factors, using well-defined concepts and measures and seeking a comprehensive integrated models that incorporate a global view.

Chapter 9 entitled, “Constructive Design Environments: Implementing End-User Systems Development” by John Gammack of Murdoch University (Australia) develops the case for centering definitions and process-flows on end users in their active situations. The chapter examines the potential for basing integrated information systems development upon the constructive and evolutionary processes in client context. The chapter considers case studies and representative situations at the levels of full application design, workflow definition and enterprise wide-development.

Chapter 10 entitled, “An Information Systems Design Framework for Facilitating TQM Implementation” by Nazim Ahmed of Ball State University and Ramarathnam Ravichandran of Design Systems (USA) provides a framework for information systems design for total quality management (TQM) implementation. The framework consists of three phases: tasks, analyses of communication effectiveness, and appropriate IS component inventories. The authors then apply their framework to a hypothetical example of a large manufacturing firm.

Chapter 11 entitled, “Methodology of Schema Integration for New Database Applications: A Practitioner’s Approach” by Joseph Fong of City University of Hong Kong, Kamalakar Karlapalem of Hong Kong University of Science and Technology and Qing Li and Irene Kwan of Hong Kong Polytechnic University (China) presents a practitioner’s approach to integrating databases and evolving them to support new database applications consisting of a joint bottom-up and top-down approach.

Chapter 12 entitled, “CMU-WEB: A Conceptual Model for Designing Usable Web Applications” by Akhilesh Bajaj and Ramayya Krishnan of Carnegie Mellon University (USA) proposes a three-dimensional classification space for Web applications, consisting of a degree structure of pages dimensions, a degree of support for interrelated events dimension and a location of processing dimension. The chapter then proposes a usability design metric for Web applications. The authors use CMU-Web, a conceptual model used to design Web applications as a way to measure these dimensions.

Chapter 13 entitled, “The Effects of Using a Triangulation Approach of Evaluation Methodologies to Examine the Usability of a University Website” by Dana Smith, Zhensen Huang, Jennifer Preece and Andrew Sears of the University of Maryland-Baltimore County (USA) report on the results of a study used to evaluate the current University of Maryland Baltimore County Web in order to identify problems to be addressed in the redesign project. With the analysis of the results collected from gathering test data, observing users and interviewing individuals from the campus, the authors were able to identify problems that could be addressed. Furthermore, the authors demonstrated the value of using a triangulation approach to devise these results.

Chapter 14 entitled, “Adaptive Web Representation” by Arno Scharl of Vienna University of Economics (Austria) classifies hypertext applications into three categories of information and their corresponding interface representation: context of documents, primary navigational system comprising links between and within the documents and supplemental navigational systems such as indexes, trails or guided tours.

Chapter 15 entitled, “Usability: Changes in the Field – A Look at the System Quality Aspect of Changing Usability Practices” by Leigh Ellen Potter of Griffith University (Australia) examines traditional usability testing and compares it to user-centered design practices focusing on the resultant quality of the information system. The author examines the literature surrounding each approach and offers comparisons to a case study of a large Australian organization utilizing both measures. The chapter reports the experiences of developers and users within the organization and discusses the perceived quality of systems developed using both approaches.

Chapter 16 entitled, “Facilitating End User Database Development by Working with Users’ Natural Representations of Data” by Valerie Hobbs and Diarmuid Pigott of Murdoch University (Australia) presents two case studies in which the first stage of the development process was completed entirely by the end user, making use of their own understanding of the dataset, the problem domain and the tools that were familiar to them. An IT expert then facilitated the conversion of the dataset to a relational database with the participation of the end users. The chapter reports on the benefits of this method of database development.

Chapter 17 entitled, “User Developed Applications: Can End Users Assess Quality?” by Tanya McGill of Murdoch University (Australia) investigates the ability of end users to assess the quality of applications they develop. The chapter confirms that there are differences between the system quality assessments of end user developers and independent expert assessors. The results suggest that end users with little experience might erroneously consider the applications they develop to be of high quality. The authors then discuss the implications of their findings.

Chapter 18 entitled, “Toward an Understanding of the Behavioral Intention to Use a Groupware Application” by Yining Chen and Hao Lou of Ohio University (USA) provides an illustration of expectancy theory, using the case of a groupware application. The chapter shows that expectancy can be applied early in the design phase of systems development to provide a better indication of a user’s intention to use a groupware application. The authors then discuss ways to maximize systems success.

Understanding human factors in information systems design and management is essential to achieving and maintaining optimal information systems. The chapters in this book represent the best research currently available on end users and human computer interaction. They address the critical issues of what role end users should play in database development, whether or not end user perceptions of their own developments are accurate and how to motivate users to implement specific practices. The chapters represent university and university settings and cover topics ranging from Web site usability to groupware use. These chapters will prove essential to academics, researchers and practitioners alike who will benefit from the insightful theoretical discussion as well as practical examples and useful case studies illustrating the concepts discussed. This book is a must-have for all those interested in understanding and applying the most up-to-date research and practice in end user computing and human computer interaction.

IRM Press
January 2002

Chapter 1

Towards User-Oriented Control of End-User Computing in Large Organizations

Neil McBride

De Montfort University, United Kingdom

A. Trevor Wood-Harper

University of Salford, United Kingdom &

University of South Australia, Australia

Control is a major issue in end-user computing. The migration of responsibility, resources and authority from IT departments to user departments is frequently seen as a loss of power by the IT departments and an erosion of cost control by senior management. Reactions to this situation tend to focus on technology and formal control mechanisms. This paper contrasts such an IT-oriented view with a proposed, alternative user-oriented view. An IT-oriented view of EUC focuses on the problems it causes, the technology it requires, the methods that should be used and the means of limiting, controlling and standardizing. An user-oriented view of EUC focuses on the problems it solves, the user's task and the organizational environment. The paper advocates a shift in EUC research away from the technology and the IT issues towards the political, social and cultural issues associated with the users. EUC problems are, in the main, organizational problems requiring a research approach which addresses dynamic issues emerging over a period of time. As a basis for such research, the paper proposes a dynamic model for EUC in which the progression of EUC within an organization is visualized as a series of inference loops.

INTRODUCTION

The advent of end-user computing (EUC) catalyzed by increasingly simple technology and increasingly sophisticated users has brought with it both solutions to problems within the information technology (IT) departments and new problems. While providing one solution to the so-called applications backlog, it has created new problems of control for the IT department, which, in some cases, has led IT departments to avoid supporting EUC, and consider outsourcing end-user training, the support of PCs and networks and the help desk. EUC has led to an increase in the workload of the IT department, a growing application backlog as EUC systems require repair and support from the IT department, and increasing conflict between users and the IT department as the IT department seeks to rein in the uncontrollable proliferation of EUC.

At the heart of these problems lies the issue of control of EUC. Robson (1997, p. 382) refers to EUC as user-controlled computing. Responsibility, resources and authority over IS move away from IT departments into user departments. EUC within the organization is affected by politics, culture and power within the organization. Reasons for the proliferation of EUC may include the wish to wrest control of IT from the IT department and to concentrate power within particular departments. The shift of control over IT resources to user departments has been associated with the duplication of computer applications, incompatibility and lack of integration, and low quality systems (Taylor et al., 1998). However, over-control of EUC by the IT department leads to alienation of end-users and conflict (Beheshtian & Van Wert, 1987). Many organizations consider the solution to the lack of control of EUC to be the exertion of more control from the center. This IT-centered view of EUC sees EUC as a problem to be solved through standards, auditing, and financial control mechanisms which seek to make end users behave like IT professionals. Literature within the EUC field emphasizes the need for management of EUC by the IT department through the use of restrictions on users (Alavi, Nelson and Weiss, 1988; Behseshtian and Van Wert, 1987; Ngwenyama, 1993; Taylor et al., 1998).

This paper firstly defines the IT-oriented approach to EUC control based on published research (Taylor et al., 1998). This is then contrasted with a user-oriented approach to EUC. A research agenda for studying EUC development from a user-oriented point of view is developed and supported by a model. It is concluded that research in EUC needs to address user motivations and the dynamics of end-user development within an organization.

AN IT-ORIENTED APPROACH TO EUC

If inadequately managed, EUC may become a source of problems. Valuable resources within IT are diverted to support amateur users who produce badly-written systems of no strategic value. There is a constant battle to halt the proliferation of various and incompatible platforms, to control spending, and to deal with problems caused by bad design and nonprofessional approaches to application development.

The case study described in Figure 1 illustrates some of the problems. An IT department focused on mainframe and large systems alienates the individual end-user whose needs are not being met. The availability of cheap PC technology provides a means for those users to take control of their computing needs. Through word-of-mouth and by example, the use of small packages spreads throughout the organization. IT finds itself faced with needs for support from a whole class of users who were previously excluded from organizational computing. The IT department is ill-prepared to meet the needs of the changing customer base. End-users consequently seek support elsewhere including non-IT departments and informal networks (Govindarajulu and Reithel, 1998).

Figure 1: Case Study: BIS Health Care

BIS Health Care is a wholly owned subsidiary of BIS UK. Based at Swindon, it is the European center for pharmaceutical manufacturing, employing 600 people on four sites. The IT Department consists of three sections:

1. Operations. Deals with running of the mainframe, management of user authorizations, and support of mainframe applications.
2. Database. Manages the Health care customer and product databases.
3. Information Center. Provides user support for in-house mainframe applications and user-programmed mainframe applications, particularly user-programmed database queries. Limited support of some PCs for technical users in the Research and Development areas has been provided in the past.

IT operations centered around the support of a mainframe running DOS /VSE.

In the last year as a result of the reorganization of European operations of BIS, the mainframe has been moved to Reading. This has catalyzed a move towards increasing use of PCs, which is causing serious problems for the Information Center. The nature of the average user has changed. Rather than in-depth technical support for a few specialist packages, broad support is now required for users with limited computer knowledge. The number of calls to the Information Center has increased dramatically, leaving the staff over-stretched.

The number of PCs within BIS Health Care is unknown. Many departments have purchased PCs for staff on internal capital budgets without the knowledge of the IT department. Requests by the IT department for information on numbers of PCs have been ignored, and new PC users are 'emerging from the woodwork almost daily'.

Relationships between users and the IT department are difficult. One user described the IT Department as 'a bunch of user un-friendly, customer un-focused techno-freaks.'

The response of IT to such loss of control may be to adopt an authoritarian attitude by creating organizational rules for the use of PCs; for example, removing hard disks from PCs on client-server networks so that users must store applications on a central server; placing restrictions on the purchasing of computers; blocking access to organizational databases unless the EUC applications which may derive data from these databases have been audited and approved; and refusing to support nonstandard systems and software. Such IT-oriented solutions arise from the perception that the control of EUC is an IT problem. It is not seen that the IT department's problem may be the user's solution. Discussion of an EUC research study will further illustrate this.

The questions addressed in Taylor et al. (1998) concern some of the problems of EUC and conclude that part of the solution lies in the adoption of a systems development methodology by the end users. Based on case studies of 34 organizations, they identify duplication of effort, low quality of end-user developed systems, and the lack of training of end-user developers as key problems. The research focused on IT departments and interviewing IT staff about EUC. This work provided a widespread and intensive survey of EUC within UK organizations from an IT viewpoint. It highlights the IT-oriented focus of EUC research.

The questions addressed in this work concerned the nature of EUC development and included:

- How is the development and maintenance of end-user computing applications carried out?
- How is the quality of end-user computing projects assured?
- How are end-user computing projects supported by the IT department?

These questions reflect the concerns of the IT professionals which may not be those of the users. The researchers used the case study material to identify several strategies for using information systems methodologies in the development of end-user computing projects: End-users should develop and maintain systems to the same standards as IT departments. They should adopt a 'cut-down' version of the IT department's methodology, tailored with the help of IT advisors to be contingent with the end-user department's needs. There is an underlying assumption that the solution to EUC problems is the same as that for IT department computing problems, namely the application of methods and standards: EUC problems will be solved if end-users become closet IT professionals. The advantages given for the adoption of methodologies in EUC are the reduction of duplication of effort and maintenance problems, the improving of quality, security and recovery, and the aligning of IT department and EUC systems (Taylor et al., p93). These may have been seen as advantages from the point of view of IT who are interested in how computing is done. They may not be of relevance to users who are interested in what is done and why.

In summary, an IT-oriented view of EUC focuses on the problems it causes, the technology it requires, the methods that should be used and the means of limiting, controlling and standardizing. A good outcome from EUC is defined in terms of the technical quality of the resulting application, the extent to which it follows the rules laid down by IT and the extent to which it integrates with IT's technology strategy.

A USER-ORIENTED APPROACH TO EUC

If an IT-oriented view of EUC focuses on the problems that EUC causes, a user-oriented view focuses on the solutions it provides. Control remains with the users and EUC problems are treated as organizational problems, not IT problems. For example, the duplication of applications and the redundancy of data that is often associated with EUC may be seen not as a result of a lack of IT standards and methods to be resolved by the imposition of control by IT, but rather as a symptom of an organizational problem. System duplication indicates organizational failure, not lack of involvement by IT. In one hospital, duplicate systems emerged as a result of organizational culture and politics: different specialties wished to assert their autonomy through the development of their own applications, and the control of their own data, raising barriers with other specialties and management (Hackney & McBride, 1995). Duplication of effort may arise from the hierarchical structures prevalent in organizations. Solutions to the duplication of systems may involve the restructuring of the organization and the establishing of better communication channels.

End-users tend to develop computer systems to solve problems of immediate concern to them. These immediate problems need rapid solutions, so time is a significant factor. End-users cannot wait for IT to produce systems (Fahy and Murphy, 1996). End-users may be uncertain as to the solution to the problem and wish to experiment. EUC may involve establishing information needs in order to reduce task uncertainty (Blili et al., 1998). The focus of the end-user is on the goal and not the means to the goal. In user-oriented EUC, quality considerations should focus on the quality of the solution and the resulting benefits rather than the quality of the tool produced to achieve that solution. An IT-oriented focus on code quality, documentation, backup and recovery misses the point of the end-user system.

End-user training is a key issue in EUC. Igbaria and Zviran (1996) suggest that computer experience and training are key to effective EUC. Ngwenyama (1993) recognises the problem of end-user competence and proposes a solution based on collaborative action learning. Zinatelli et al. (1996) identify computer experience and computer training as key factors in encouraging EUC sophistication. While there is little argument about the importance of training and experience, the nature of that training is open to debate. Some authors advocate an IT-oriented view which focuses on training in the technology, methods and standards. Taylor et al. (1998)

suggest training users in MicroSSADM, which is a reduced and simplified version of SSADM (Structured Systems Analysis and Design Method). Other authors advocate training in tools and IS concepts (Alavi et al., 1988; Beheshtian and Van Wert, 1987). User-oriented EUC training should focus on identifying problems and solutions and evaluating potential IT tools. Rather than training that seeks to turn an end-user into an IT professional, training should focus on making end-users better at their tasks through the effective use of information systems, whether these are existing systems or are built by the end-user. IT issues such as database management, backup and recovery should be handled automatically by the end-user computing tool or handled sensitively in the background by IT professionals.

The use of a systems development methodology by end-users may be regarded as an attempt to impose an IT culture on end-users. This culture may be foreign to the users (Ward and Peppard, 1996; Peppard and Ward, 1999). An IT-oriented view of the advantages of the use of a methodology in EUC may be interpreted by users as reasons for not using a method. Table 1 offers a possible user view of each of the advantages given for the suggested use of methodologies by Taylor et al. (1998).

Table 1: Contrast Between IT's View and the User's View of the Use of Methodologies in EUC.

IT View	User View
Reduces duplication	Removes my autonomy and ownership of the data.
Reduces difficulty of maintenance	Removes dependency on me as the system expert, reduces the extent to which I am needed to understand the problem and my creative solution to it.
Improves quality	Reduces creative input, reduces my ability to develop an evolving solution which reflects who I am (my role in the organization) and my ability to develop my skills.
Improves security	Reduces accessibility of system, reduces my ability to gain kudos by spreading my clever ideas around the department
Improves backup and recovery	Increases time wasted on non-essential, technical activities which I don't want to worry about because they are not part of the problem I am working on.
Aligns IT department and EUC	Allows IT to interfere with the way I work, increases IT's power and control which I am trying to break free of, reduces my independence.

A thesis of this paper is that the control of EUC should remain with the user, and that IT involvement should be limited to providing advice, perhaps through the mechanism of information centers (Gunton, 1988; DeVargas, 1989; Khan, 1992), only if requested. Attempts by IT to control EUC and enforce an IT-oriented approach are likely to generate resentment and fail. Alavi et al. (1988) suggest that EUC control should be enforced through line management and not by IT personnel. Beheshtian and Van Wert (1987) argue that, while IT should suggest standards and controls, it cannot be expected to enforce them since it is unlikely to have the authority or the resources. If IT is to be involved in EUC it may be done by relinquishing control of IT staff to the users. Govindarajulu and Reithel (1998) found that 62% of organizations in their survey had decentralized support for EUC by placing IT staff in user departments. In user-oriented EUC, control of computing activities is taken away from IT which signals that EUC is an organizational issue, not an IT issue.

The removal of EUC control from IT, or any centralized authority, may enhance the risk of complications - system redundancy, data duplication, lack of data integrity. However, this may bring with it increased creativity, the extension of organizational knowledge, and greater opportunity of the creation of strategic information systems (Davenport, 1994; McBride et al., 1997). Effective solutions may be embedded in everyday experience and local knowledge; open experimentation by end-users should be encouraged; ideas should emerge from deviations from standards and from initiatives outside IT's development agenda (Ciborra, 1994).

In summary, a user-oriented view of EUC focuses on the problems it solves, the user's task and the organizational environment. Technology is provided unobtrusively as a background tool supporting the end-user in delivering business benefit. A good outcome from EUC is defined in terms of the business quality of the solution provided by the end-user (the extent, for example, to which it reduces costs, increases efficiency and increases customer satisfaction), and the extent to which it contributes to business goals.

RESEARCH QUESTIONS FOR USER-ORIENTED EUC

We argue that a reframing of EUC research is required. Both the subject and the method of research need to change. EUC needs to be viewed from a user's point of view as well as an IT point of view. While IT-oriented research is important, too much of the survey work within information systems has solicited only the views of IT practitioners and largely ignored the views of users (Galliers et al., 1994). While IT-oriented research on EUC focuses on IT problems (Taylor et al., 1998), user-oriented EUC focuses on end-user's needs (Fahy and Murphy, 1996). Important areas of research concern the user's motivation, the nature of user tasks, and the

role of the user within the organization. The IT-oriented research questions of Taylor et al. (1998) are replaced by user-oriented questions:

- What has motivated the user to start EUC?
- What are the user's objectives in doing some programming?
- What is the user's attitude to computing, to the IT department, to information?
- What is the primary focus of the problems the end user is tackling?
- What are the problems that EUC solves?
- How do those problems relate to the business's corporate objectives?
- Why do end-users ignore standards and guidelines?

Research in EUC should focus on motivation, attitudes, the development of experience and the triggers which cause or promote end-user computing developments. EUC emerges over time. Therefore, a research approach is required which addresses the dynamic issues and discovers the emerging patterns and influence on the end-user's activities and attitudes. Static studies based on surveys or interviews will not reveal the complex and developing interactions which change the way computing is carried out within an organization. Longitudinal studies are required which build up a history of the development of EUC within an organization and demonstrate the emerging, cyclical patterns (Weick, 1979). Static studies, even when taking a case study approach (Taylor et al., 1998; Zinatelli et al., 1996) may not provide the rich detail required to interpret EUC development.

EUC arises from the complex relationships between groups, individuals and technologies. The motivation for EUC needs to be determined and the effect of EUC on user motivation analyzed. EUC may increase satisfaction in work through providing self-expression, self-determination and intrinsic job satisfaction. Users can influence job design and determine their own information requirements. They can increase their skills, deriving satisfaction from the expression of those skills and from self-expression. It can be argued that EUC leads to greater job variety, complexity, autonomy and responsibility, which may lead to greater job satisfaction (Katz and Kahn, 1978).

Interpretive studies are required which seek to examine the dynamics of EUC. These studies must ask how end-users produce change in their environment and identify areas of organizational change requiring further attention. The user of IT in mediating such change needs to be examined. EUC studies must understand how end-users interpret their organizational environment and impose structure on it, how they differentiate between figure and ground (Weick, 1979), which is between what is seen as interesting, important and worthy of focused attention and the background information that is assumed, taken for granted or ignored. The use of EUC may help in retaining and formalizing the end-users' interpretive structures; and their understanding of their roles, processes and customers.

A DYNAMIC MODEL OF EUC

The progression of EUC within an organization may be visualized as a series of inference loops (Weick, 1979) which develop over time. Effects within loops are amplified and small factors may take on great significance as EUC evolves. The following describes a theoretical model which seeks to explain the interactions which influence EUC within an organization, described in terms of inference loops. The attributes describe discrete events; the arrows connect events and represent influence. Weick (1979) also describes these events as variables which can have a variety of values. These inference loops bear some similarity to the causal maps used in comprehensive situational mapping (CSM) (Offodile and Acar, 1993; Georgantas and Acar, 1995). In CSM, nodes represent influencing functions or attributes and arrows represent influencing vectors and are given a signed magnitude. However, in CSM, causal maps support decision making, whereas Weick's inference loops support sense making in complex social situations within organizations.

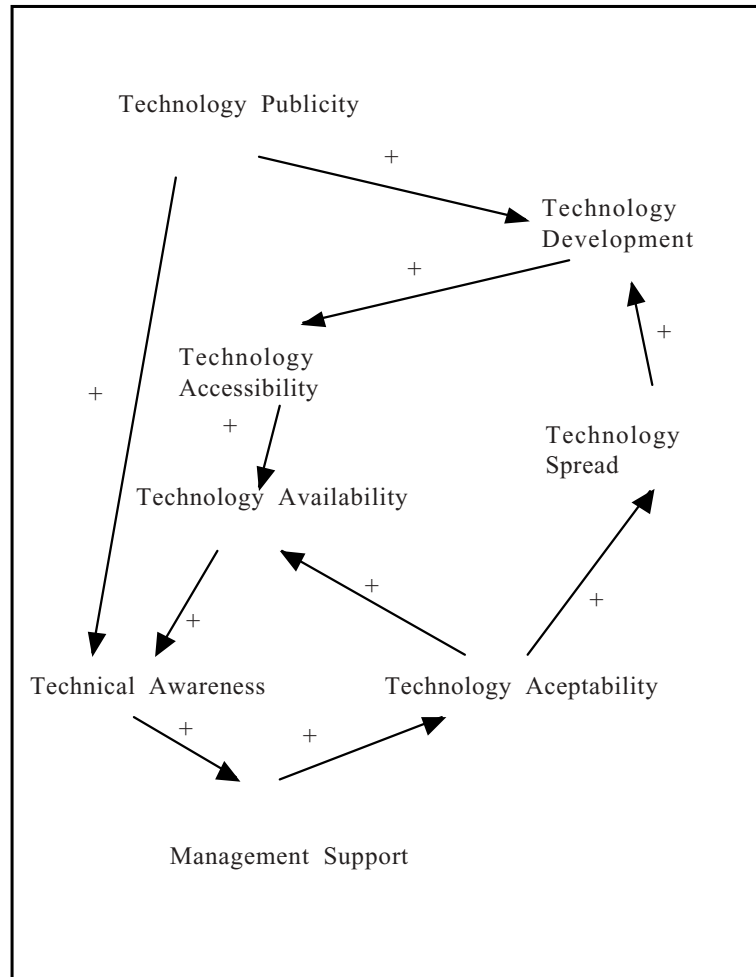
TECHNOLOGY IMPROVEMENT

A key element of EUC is the availability of the technology. EUC requires cheap technology that is easy to use. Figure 2 illustrates possible inference loops based on the following attributes:

- **Technology Accessibility.** The ease of procurement and use of IT, influenced by the low cost of workstations, the ease of implementation, and the ease of end-user system development.
- **Technology Availability.** The extent to which the end-user has access to PCs and workstations.
- **Technology Awareness.** The knowledge that the end-user has of what IT is available and how it can influence her work tasks.
- **Technology Acceptability.** The extent to which the use of IT is an accepted part of work practice and is embedded in the end-user's tasks; the extent to which the use of IT is a natural element of the end-user's role and the extent to which IT use is an organizational norm.
- **Management Support.** The extent to which management encourages the use of IT by their subordinates and the extent to which they encourage end-user developments and initiatives. This will be influenced by the management's awareness of the technology.
- **Technology Spread.** The extent to which IT spreads with the organization. This might be examined by looking at changes in the number of users that have PCs or workstations on their desk.

Figure 2: Technology Development

(+ indicates one attribute causes increase in another, - indicates one attribute reduces another.)



- **Technology Development.** The way the technology is used within the organization, the maturity of information systems support, the development of the technology platform, the provision of better development tools.
- **Technology Publicity.** The extent to which the organization is exposed to publicity about changing technology in the popular and trade press, through word of mouth and through supplier advertising including supplier visits and trade fairs.

The availability of the technology is necessary but not sufficient for the uptake of EUC. There must be group acceptance of the technology and the establishing of an environment in which the use of computers is seen as socially acceptable. Social acceptability may emerge from management support, strengthened by the rules, norms and interpretations placed on the technology. We must ask: how does the management interpret the role of information technology within the organization and its use by end-users?

Figure 2 illustrates the positive influence of the attributes on each other. For example, increased technology availability and technology publicity may lead to increased technical awareness and consequently increased management support. It should be noted that the figure also suggests a decrease in one attribute will lead to a decrease in another. Thus reduced technology availability and reduced technology publicity may lead to reduced technical awareness and consequently reduced management support.

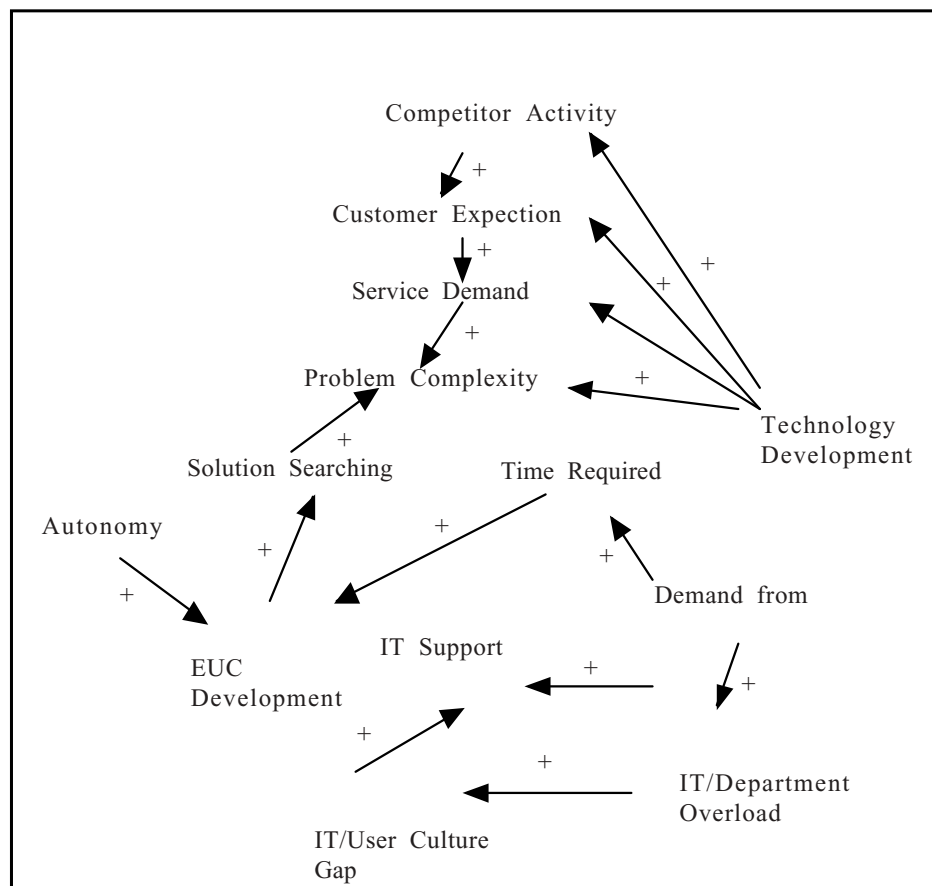
IT DEPARTMENT INVOLVEMENT

The role of the IT department is crucial to the development of EUC. Figure 3 identifies some suggested causal influences based on the following attributes:

- Demand from Users. The demand for new IT development from users as represented by development project requests, information system usage, involvement of end-users in systems development.

Figure 3: IT Department Involvement

(+ indicates that one attribute causes increase in another, - indicates one attribute reduces another.)



- **IT Department Overload.** The size of the gap between the number of requests for development, maintenance and support work from end-users and the available development resources, both staff and capital, to meet the demand.
- **IT/User Culture Gap.** The extent to which the IT and business functions within the organization are aligned in terms of strategy, organizational goals, empathy, professional respect, geographical location and knowledge of the business.
- **IT Support.** The level of IT support provided to users to enable them to fulfill their organizational roles effectively and efficiently, as perceived by the end-users themselves.
- **EUC Development.** The extent of development of end-user systems within the organizations and its departments, as suggested by the number of users involved, the amount of time spent by users in system development, the size of the resulting systems, the extent of usage of those systems and the importance of the systems to the organization.
- **Autonomy.** The extent to which the end-users have control over their IT budget, the selection of IT systems, the way they carry out business processes and the outcomes of those business processes.
- **Time Required.** The amount of time required to deliver a new information system development, in terms of actual time needed, which is affected by the size and complexity of the system, and elapsed time, which is also affected by the availability of resources and the waiting period before a project can begin.
- **Competitor Activity.** The extent to which the organization's competitors are using IT to develop new services and enhance existing services.
- **Customer Expectation.** The perception of customers as to the level of service the organization should provide and the types of service. This is influenced by what competitors are providing and by their use of IT.
- **Service Demand.** The demands placed on the organization in terms of the volume, level, quality and complexity of service provided.
- **Problem Complexity.** The complexity of the problem for which end-users are developing a computer-based solution, considered in terms of number of data items, algorithm complexity, number of processes and interactions.
- **Solution Searching.** The amount of effort expended by end-users in researching the use of information systems to provide solution to business problems.

Technology improvement may lead to increased demand for IT services. This in turn may lead to IT distancing itself from the user in order to minimize the resources being directed away from major operational IT projects. However, in a dynamic environment the effect of a factor may change suddenly. For example, while initially the lack of IT support increases EUC activity since demands for systems are not being met by IT, when the end-user subsequently hits development problems, the

absence of IT support may act as an inhibitor of EUC since the user cannot proceed without advice and expertise which is not forthcoming.

The motivation for EUC lies in the need for end-users to overcome problems which affect their day-to-day processes. The complexity of the problem leads to an increase in EUC as the problem solvers seek to develop solutions which reduce complexity and make the operational situation manageable. Problem complexity may be influenced by the availability of improved technology which leads to greater demands from customers. End-users require rapid solutions to problems. Often time limitations motivate the user to undertake her own system development. Both problem complexity and IT overload may be seen as increasing the time needed for a problem to be solved. Increased waiting may increase the motivation to carry out EUC.

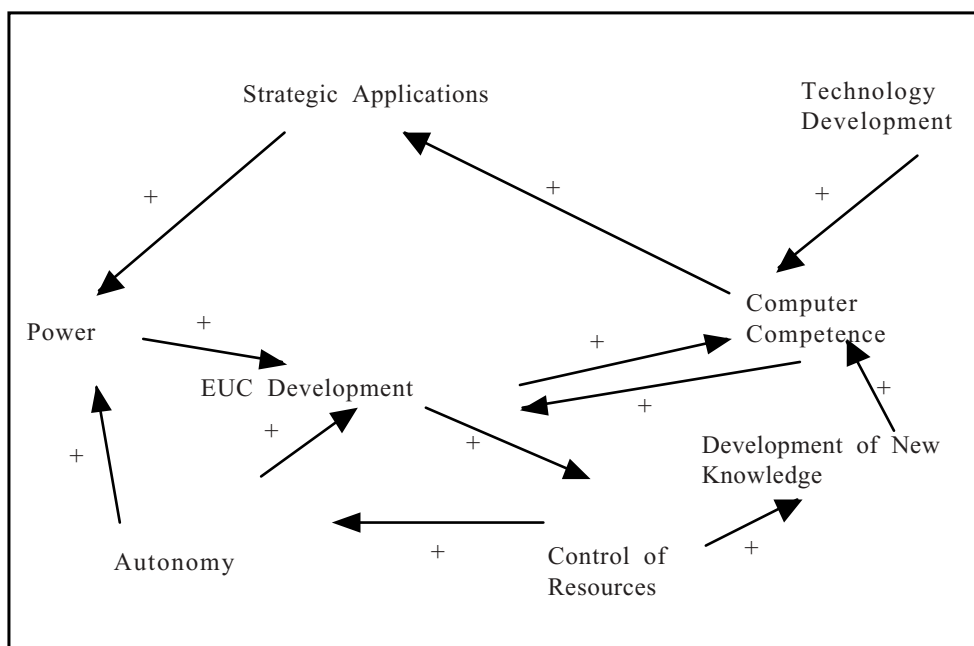
POWER DISTRIBUTION

A third series of inference loops (Figure 4) hypothesizes the effect of EUC on power. In addition to the attributes described above, the following attributes are suggested:

- **Control of Resources.** The extent to which the end-user controls the hardware and software platforms and applications. The extent to which the end-user controls the development and usage of local computer systems.

Figure 4: Power Distribution

(+ indicates one attribute causes increase in another, - indicates one attribute reduces another.)



- **Development of New Knowledge.** The rate at which the end-user takes on new knowledge and develops new skills through attending training courses and developing new systems.
- **Computer Competence.** The overall level of IT understanding of the users as shown through the usage and development of information systems.
- **Strategic Applications.** The number of applications built by the end-user which have a significant effect on the organization's business success. The extent to which particular applications built by the end-user are influential within the organization and have high visibility.
- **Power.** The perceived influence of end-users on the distribution of resources, the decision-making process, and the strategic direction of the organization.

Increased EUC may result in increased control of resources by the end-user. This may lead to increases in autonomy and power within the user community. Furthermore, the expansion of EUC may lead to increased computer competence. This may in turn lead to the development of new strategic applications by end-users which may increase their power base within the organizations.

RESEARCH STRATEGY

The above model is based on our understanding of the important issues in EUC, drawn from both the literature (Alavi et al., 1988; Hackney and McBride, 1995; Taylor et al., 1998) and our own experience. As such, the model is untested: research is needed to develop it. Since the model describes the influences of factors on EUC over a period of time, the progression of these inference loops may be best studied through case studies developed over time to provide an historical analysis of the progression of EUC. Interviews should be conducted with end-users at intervals over a period of time. The gathering of local knowledge, local stories and local meaning (Colville et al., 1999) will enable an understanding of these phenomena to be built up. We are not advocating that the attributes within the inference loops be necessarily treated as quantitative measurables. They may be used as conceptual guides, sensitizing the researcher to themes that should be developed in the storytelling. We see this as a way of interpreting and understanding processes, rather than developing objective measures. However, we recognize that the model of EUC may be investigated quantitatively by seeking to measure the change in each variable over time.

These studies need to recognize the importance of external influences on the development of EUC, the intimate link between EUC development and organizational dynamics, and the effect of feedback. As a result of case study research, these models may provide a practical basis for directing resources towards EUC, developing an appropriate organizational culture and optimizing the use of information and communication technologies within the organization.

CONCLUSION

The advent of Internet technology and the development of Intranets as the basic information infrastructure for an increasing number of organizations may accelerate a sea-change in approaches to IT management. Intranets offer end-users increased freedom from IT-oriented control of organizational computing. Instead of being dependent on centrally defined menus and systems, end-users are free to select the systems they want and to develop their own personalized information environments. End-users may develop home pages which contain information sources which are relevant to their organizational roles, rather than using company-wide systems which may not be of significant value.

The scale of this change may catalyze such a change in information management that end-user computing becomes the dominant form of organizational computing. A user-oriented view of EUC may become essential for both researchers and practitioners. The technological view of EUC control centered around standards, methods and technological audits may not be an appropriate approach to a series of problems which concern organizational context, culture and politics.

EUC problems are, in the main, organizational problems requiring a research approach which addresses dynamic issues emerging over a period of time (Watson and Wood-Harper, 1996). EUC research must draw out the organizational issues which drive EUC. A user-oriented view may enable a focus on user tasks and problems and the way IT can serve the user and solve the user's problems, not on the technology and the way the user can serve the technology. User-oriented EUC research may lead to alternative approaches to IT development and support. This may involve the use of component technology, the development of tailorable, evolving systems, and the use of disposable software to solve immediate problems. Flexibility and tailorability may be more important than structure and method. User-oriented EUC should be judged on business value and problem-solving success, not methodological rigor. New EUC research should be business-focused rather than technology-focused, understanding the motivations for EUC and the nature of successful outcomes.

In order to gain empathy and understanding, IT departments must view the development of EUC within an organization from the user's point of view. If the IT department understands the user's motivation, both explicit and tacit, it may be able to provide help both technically and managerially. That help must be anticipatory and unobtrusive. This paper identifies a research need, the outcome of which will help IT departments to understand EUC and respond appropriately.

REFERENCES

- Alavi, M., Nelson, R.R., & Weiss, I.R. (1988). Managing End-User Computing as a Value-Added Resource. *Journal of Information Systems Management*, Summer 1988, 26-35.
- Behshatian, M., & Van Wert, P.D. (1987). Strategies for Managing User Developed Systems. *Information and Management*, 12, 1-7.
- Blili, S., Raymond, L. & Rivard, S. (1998). Impact of Task Uncertainty, End-User Involvement and Competence on the Success of End-User Computing. *Information and Management*, 33, 137-153.
- Ciborra, C. (1994). The Grassroots of IT and Strategy, in *Strategic Information Systems: A European Perspective*, Ed, Ciborra, C & Jelassi, T, John Wiley, 3-24.
- Colville, I.D., Waterman, R.H. & Weick, K.E. (1999). Organizing and the Search for Excellence: Making Sense of Times in Theory and Practice. *Organization*, 6(1), 129-148.
- Davenport, T.H. (1994). Saving IT's Soul: Human Centered Information Management. *Harvard Business Review*, March-April 1994, 119 - 131.
- Devargas, M. (1989). *Introducing the Information Center*. NCC Blackwell.
- Fahy, M. & Murphy, C. (1996). From End-User Computing to Management Developed Systems. Proceedings of the 4th European Conference on Information Systems, Ed, Coelho, J.D., Jelassi, T., Konig, W., Krcmar, H. & O'Callaghan, R., 127 - 142.
- Galliers, R.D., Merali, Y. & Spearing, L. (1994). Coping with Information Technology? How British Executives Perceive the Key Information Systems Management Issues in the Mid-1990s. *Journal of Information Technology*, 9(3), 223-238.
- Georgantzias, N.C. and Acar, W. (1995). *Scenario-Driven Planning: Learning to Manage Strategic Uncertainty*. Quorum Books.
- Govindarajulu, C. & Reithel, B.J. (1998). Beyond the Information Center: An Instrument to Measure End-User Computing Support from Multiple Sources. *Information and Management*, 33, 241-250.
- Gunton, T. (1988). *End User Focus*. Prentice Hall.
- Hackney, R. & McBride, N.K. (1995). The Efficacy of Information Systems in the Public Sector: Issues of Context and Culture. *International Journal of Public Sector Management*, 8(6), 17 - 29.
- Igbaria, M. & Zviran, M. (1996). Comparison of End-User Computing Characteristics in the US, Israel and Taiwan. *Information & Management*, 30, 1-13.
- Katz, D. & Kahn, R.L. (1978). *The Social Psychology of Organizations*. John Wiley, Chichester.

- Khan, E.H. (1992). The Effects of Information Centers on the Growth of End-User Computing. *Information & Management*, 23, 279-289.
- McBride, N., Lander, R. & McRobb, S. (1997). Post-Modernist Information Management. *Proceedings of the 7th Annual Business Information Technology Conference*, Manchester Metropolitan University, November 5-6 1997.
- Ngwenyama, O.K. (1993). Developing End-User's Systems Development Competence. *Information & Management*, 25, 291-302.
- Offodile, O.F. & Acar, W. (1993) Comprehensive Situation Mapping for Robot Evaluation and Selection. *International Journal of Operations and Production Management*, 13(1), 71-80.
- Peppard, J. & Ward, J. (1999). 'Mind the Gap': Diagnosing the Relationship Between the IT Organization and the Rest of the Business. *Journal of Strategic Information Systems*, 8, 29 - 60.
- Robson, W. (1997). *Strategic Management and Information Systems*. Pitman, London.
- Taylor, M.J., Moynihan, E.P & Wood-Harper, A.T. (1998). End-User Computing and Information System Methodologies. *Information Systems Journal*, 8, 85-96.
- Ward, J. & Peppard, J. (1996). Reconciling the IT/ Business Relationship: a Troubled Marriage in Need of Guidance. *Journal of Strategic Information Systems*, 5, 37-65.
- Watson H. & Wood-Harper A. T. (1996). Deconstructing Contexts in Interpreting Methodology. *Journal of Information Technology*, 11(1), 36-51.
- Weick, K. (1979). *The Social Psychology of Organizing*. McGraw-Hill, New York.
- Zinatelli, N., Cragg, P.B. & Cavaye, A.L.M. (1996). End-User Computing Sophistication and Success in Small Firms. *European Journal of Information Systems* 5, 172-181.

Chapter 2

On-Line User Interaction with Electronic Catalogs: Language Preferences Among Global Users

Aryya Gangopadhyay & Zhensen Huang
University of Maryland Baltimore County, USA

In this paper we study the behavior and performance of bilingual users in using an electronic catalog. The purpose of this research is to further the knowledge required for building electronic commerce systems that operate in multiple languages in global settings. We describe a bilingual electronic catalog that can be used by online retailers for selling products and/or services to customers interacting in either English or Chinese. We investigate into the nature of user interactions in multilingual electronic catalogs. We have defined three different groups of users: only Chinese speaking, only English speaking, and bilingual. We are specifically interested in investigating into the language preferences of the third group of users. In order to test language preferences, we have selected two types of products: office supplies and ethnic food. We hypothesize that bilingual users will exhibit differential language preferences for the type of products and the tasks performed in using the electronic catalog. Furthermore, learning curves and interaction effects are also tested. Three different task categories have been designed: browsing, directed search, and exact matches. In the first case, the user is a general browser who is looking for what is available in the catalog. In the second case, the user is looking for a class of products but is unsure of the exact item. In the third case the user knows exactly what item he/she is looking for. We propose to test the efficiency of usage by measuring the time

as well as studying the path followed by the user in retrieving product information. This research will shed light on the important issue of designing multilingual electronic catalogs for both local and global applications.

One of the major challenges facing organizations involved in electronic commerce today is to organize and summarize information in such a way that end-users can effectively and efficiently search for and analyze relevant information. Users can look for both structured as well as unstructured information in a system designed for electronic commerce. An example of structured information is the price of a specific product. Unstructured information, on the other hand, is one that is not well specified, or can have multiple specifications. For example, the user may be looking for spices for cooking a shrimp dish, where they can choose from a number of options, may have individual preferences¹ for the selection of spices, and may not know exactly how the information can be found in the system.

The problem of finding relevant information is exacerbated in global information management, especially in global electronic commerce. While globalization is presenting new opportunities for people and businesses worldwide, several challenges must be addressed in order to realize its full potential. Examples of these challenges include differences in culture and language, which can be an obstacle to unrestricted and free access of information, as well as the disorganization of the potentially precious knowledge asset. While language technology (Nirenburg, 1992; Onyshkevych and Nirenburg, 1995; Sheremetyeva and Nirenburg, 1996) is making rapid progress, much research is needed in managing and accessing multilingual information in order to reach full potential of global electronic commerce (e.g., Malhotra 1997, 1998).

The purpose of this research is to further the knowledge required for building information systems that operate in multiple languages. Specifically, we focus on studying user behavior in performing various tasks in a multilingual system. In order to study user behavior and performance in a multilingual electronic commerce setting, we have designed a bilingual electronic catalog which can be used by on-line retailers for selling products and/or services to customers interacting either in English or Chinese.

An electronic catalog is a graphical user interface that presents product and/or service information to users, typically using the World Wide Web. An electronic catalog is a key component of electronic commerce that has been used for business-to-consumer commerce as well as business-to-business commerce (Adam et al. 1998). Although the term electronic catalog might sound like an electronic extension of paper catalogs, it offers features that are far beyond those found in paper catalogs. Such features include computational services such as efficient browsing and searching, online order processing such as checking out products

using shopping carts and secure payment mechanisms, and backend processing such as integration with company databases (Segev et al. 1995). These features have extended the role of electronic catalogs to the point of being used as electronic storefronts.

With the rapid proliferation of electronic commerce both in local and global markets, there is an increasing need to provide support for internationalization such as foreign currencies, different date and time formats, sort order, and multiple languages (Broin 1999). The need for providing multilingual support is echoed by the rapid increase of non-English speaking users on the Internet. For example, it is reported that 60% of the users on the Internet will be non-English speaking by the year 2002 (Computer Economics 1999).

In this paper we describe a bilingual electronic catalog and describe its usability based on product characteristics and tasks performed by users. The electronic catalog supports two languages: Chinese and English, and may be extended to multiple languages.

The rest of the paper is organized as follows. In the next section we describe the electronic catalog and its components. Next, we design an experiment for testing user interaction with the catalog, followed by experimental design and analysis of results. The last section contains our conclusions and future research directions.

METHODOLOGY

Description of the Catalog

A prototype electronic catalog has been implemented on the World Wide Web using ColdFusion 4.0 as the front-end, which is connected to a Microsoft Access database at the backend, using an ODBC driver. The catalog is composed of two identical interfaces in two languages: English and Chinese. Following the *unified content model* (Doherty 1999), the English interface has been translated element by element into the Chinese interface, with the only difference being the order in which the products are sorted.

The purpose of using the unified content model was to eliminate any presentation bias in user preferences. The front-end interface in Figure 1 shows two language options (*English* and *Chinese*) and two separate applications (*Office Supplies* and *Food Market*). Figures 2a-2b show the second-level interface that is invoked once a user selects the *Food Market* application in the English and Chinese versions respectively. There are three modes of operations that a user can select in order to interact with the system: *browsing mode*, *searching mode*, and *matching mode*. In *browsing mode*, the user is looking at the products available

in the catalog without having any specific item in mind, which is supported by the list box “Select the category:” in Figure 2a. In *searching mode*, the user is searching for a product class without having a specific item in mind, which is supported by the list box “Select the subcategory:” in Figure 2a. In *matching mode*, the user has a specific item in mind, which is supported by the text box “Search by keyword:” in Figure 2a. Once the user selects a category, all products in that category are listed at the next level interface, an example of which is shown in Figures 3a and 3b for the English and Chinese versions respectively.

When an item is finally selected, the product information is displayed. At this point the user has the option to include the product in the *shopping cart*, continue to shop, or go back to the initial interface, as shown in Figures 4a and 4b for the English and Chinese versions, respectively.

The second level interfaces for the *Office Supplies* application are shown in Figures 5a and 5b for the English and Chinese versions respectively.

Figure 1: The Front-End Interface

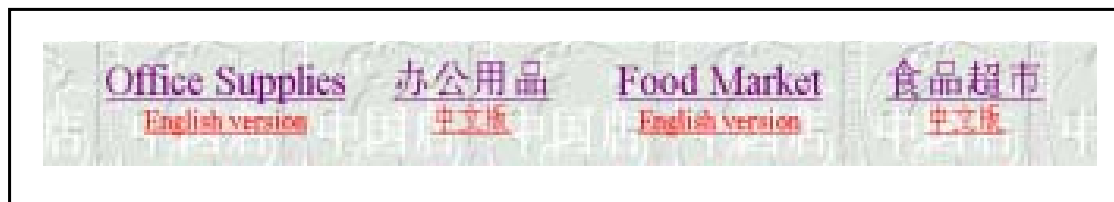


Figure 2a: Three Modes of Search in English

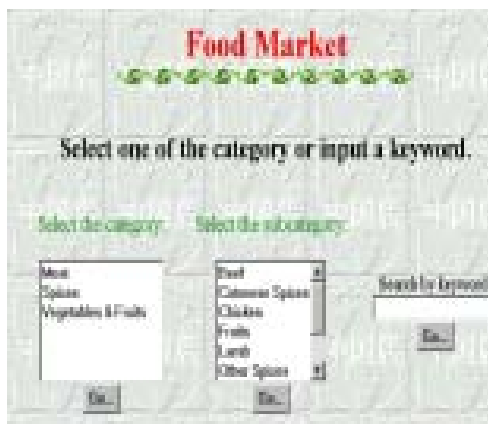


Figure 2b: Three Modes of Search in Chinese





Experimental Design

We conducted experiments with bilingual subjects speaking both English and Chinese. The subjects are randomly selected from undergraduate and graduate students, and are tested for uniformity in background and proficiency in both languages.

The purpose of the experiments is to study language preferences based on task and product characteristics. In order to achieve this objective, we have designed the experiments as follows. All subjects are required to work on the pre-defined tasks using all four applications. We define three kinds of tasks: *browsing*, *searching*, and *matching*. An example of a browsing task is as follows: “*Check out all spices that are suitable for cooking seafood.*” An example of a searching task is as follows: “*Check out the cheapest desktop computer (without monitor).*” An example of a matching task is as follows: “*purchase one chili bin sauce.*” Corresponding to these tasks, we design three kinds of search methods: *Category browsing*, *sub-category browsing*, and *keyword search*. While working on these tasks, subjects are required to use the pre-selected search method. For example, subjects are required to use category browsing for browsing tasks, sub-category browsing for searching tasks, and keyword searching for matching tasks, respectively. In addition, subjects are required to work on a fourth task, which is a little more complicated than the previous ones. In the fourth task, a subject is not given a specific search method, but can select any combination of searching method based on their needs. An example of this task is as follows: “*A recipe for Kung Pao Chicken needs the following ingredients: Chicken, peanut, cucumber, green pepper, carrots, Kung Pao Sauce. Please try to look for as many of the ingredients as you can using the Food Market catalog.*”

Thus each subject is required to finish four different tasks using all four applications (English Office Supplies, Chinese Office Supplies, English Food Market, and Chinese Food Market), which gives a total of 16 tasks. After a subject finishes an application, a short survey on their perception of usability is provided. In the survey, the subject evaluates the system (for each task) in terms of its ease of use by ranking it in a seven-point Likert Scale, with 1 being “Extremely easy” and 7 being “Extremely difficult”. Furthermore, at the end of the experiment, another short survey of user perception of language preference is provided.

In order to compare the user performance, the subjects are randomly divided into two groups. The first group performs the tasks using the Chinese language interface and then the English language interface. The second group performs the tasks by first using the English language interface and then the Chinese language interface. Two sets of task descriptions have been developed for these two groups. The content of these two sets of task descriptions is the same. The difference is that if a task appears for Group One in English, then it would be in Chinese for Group

Two, and *vice versa*. This way the sequence in which the tasks are carried out remains the same for the two groups.

Measurement of Outcomes

Four types of data are collected: the items checked out, time taken to perform a task, the path followed to perform a task, and user perception.

In order to collect the data, three modules are developed. A typical *shopping cart* in an electronic commerce environment is developed to collect the list of items checked out. A *timer* is built into the interface to measure the time it takes to perform a task in an unobtrusive manner such that it does not interfere with the user's activities. The user is neither aware of the timer nor required to complete the task in a pre-specified time limit. Also, a *path-tracing* module is integrated into the interface to record the paths as the sequence of URLs visited in completing a task.

Analysis of Results

The results are analyzed using *item analysis*, *time analysis*, and *path analysis*. In performing item analysis, two situations can arise. In the first case there is a specific list of items to be checked out for a given task. For example, for the task: "Select the cheapest desktop computer (without monitor)", the result is very specific. In this case, we can evaluate user performance by comparing the items checked out by the user against the correct list of items. In the second case, the result is not specific. For example, for the task: "Select all spices that are suitable for cooking seafood", the list of spices can vary according to the recipe used. Thus, it may be impossible to provide a "correct" answer in this case. In order to address this, we look for regularities in the item selection across all subjects. For example, let us assume that the list of items selected for a given task is more homogeneous in the Chinese language interface than in the English language interface. It then implies that the subjects have a better understanding of the products listed in the catalog that are in the Chinese language environment than they have in the English language environment.

Time analysis involves computing the average, standard deviation, minimum, and maximum time for completing the pre-defined tasks in each group. Then, the results in the two groups are compared.

Path quantification is done as follows. First, we define a concept of *cycle*: a cycle is a complete path that a subject takes for selecting one or more items. For example, if the subject selects an item using the category browsing method, we count it as one cycle. In the other case, if a subject selects the category browsing method to look for an item and fails to find it, then tries keyword searching, and gets it, the result is counted as two cycles. For each task we pre-determine the number of cycles that it takes to perform the task. For example, in order to get all the items

for the task: “Look for *Chicken, peanut, cucumber, green pepper, carrots, Kung Pao Sauce*”, the subject needs to complete at least three cycles (the order of cycles is immaterial). In the first cycle, the subject can use any method to get Chicken; in the second cycle, the subject can use category browsing to get peanut, cucumber, green pepper, and carrots; in the third cycle, the subject can use any method to get Kung Pao Sauce. Because Chicken (under meat category), Kung Pao Sauce (under spices category), and the other products (all under vegetable and fruit category) belong to different categories, there is no way to get all of them in one cycle. The second part of path analysis involves calculating the total number of cycles completed for each task. The lower the number of cycles taken to perform a task, the better the performance.

The effect of task and product characteristics on language preference will be tested using the following hypothesis.

- H1: Users of an electronic catalog exhibit language preferences based on the characteristics of the products included in the catalog.
- H2: Users exhibit language preferences based on the type of search methods they are using.

The test results can have significant implications in designing electronic catalogs for both local and global markets. There are many bilingual users in local markets and if product and task characteristics are correlated with language preferences, then the design of an electronic catalog can be dynamically modified to suit the language needs. On the other hand if users do not exhibit language preferences for product classes and/or tasks, global electronic catalogs may be designed using the *unified content model* (Doherty, 1999), which will lead to a dramatic reduction in the cost of design, rollout and maintenance of multilingual Web sites.

Time Analysis

The most important measurement in our experiment is how much time the subjects spent to finish their tasks, which is called time-spent. In our time-spent analysis, taking the Office Supplies application as an example, we compared the average time that the subjects in Group 1 spent for the Chinese language interface to that of the subjects in Group 2 using the Office Supplies application in English. Also, the average time that the subjects in Group 1 used in Office English is compared to that of the subjects in Group 2 used in Office Chinese. The same comparisons are done to the Food Market application.

The results in Table 1 are drawn from the Food application.

We use the following notation: results with capital G are significant, while with small g are not. Furthermore, Food1 indicates that the user is using the Food Market

Table 1

Application	Tasks	Group1	Group2	G1 (AVG)	G2 (AVG)	F-test	Results
Food1	Task1	Chinese	English	74.2	116.9	.007	G1<G2
Food1	Task2	Chinese	English	23.3	30.1	.000	G1<G2
Food1	Task3	Chinese	English	51.7	26.2	.0454	G1>G2
Food1	Task4	Chinese	English	195.5	198.8	.018	G1<G2
Food2	Task1	English	Chinese	58.5	50.1	.033	G1>G2
Food2	Task2	English	Chinese	29	26.8	.33	g1>g2
Food2	Task3	English	Chinese	23.3	38.7	.006	G1<G2
Food2	Task4	English	Chinese	178.2	134.5	.076	g1>g2

Table 2

Application	Tasks	Group1	Group2	G1 (AVG)	G2 (AVG)	F-test	Result
Office1	Task1	Chinese	English	71.8	64	.109	g1>g2
Office1	Task2	Chinese	English	40.9	20.4	.0007	G1>G2
Office1	Task3	Chinese	English	63.8	63	.398	g1=g2
Office1	Task4	Chinese	English	244.3	182.2	.146	g1>g2
Office2	Task1	English	Chinese	48.6	79	.0007	G1<G2
Office2	Task2	English	Chinese	61.8	41.4	.056	g1>g2
Office2	Task3	English	Chinese	56	66.3	.295	g1<g2
Office2	Task4	English	Chinese	150.2	165.2	.0156	G1<G2

application for the first time, and Food2 indicates that the user is using the Food market application for the second time in a different language than the first time.

From above, we can get the following conclusion:

- In food1, the Chinese version required less time than the English version except in the keyword search. All the results are significant.
- In food2, the Chinese version required less time than the English version except in the keyword search. Two of the results are significant.

The longer time-spent in keyword search may be because in the Chinese version, the subjects had to input Chinese characters for keyword search, which is a really time-consuming process.

The results in Table 2 are drawn from the Office application. The same notation applies as in the case of the Food market application.

From above, we can get the following conclusion:

- In office1, the Chinese version required more time than the English version except in the keyword search. However, only one of the results is significant.

The reason that almost all users exhibited the same time in keyword search is that the task for the English keyword search is more complicated than the Chinese task, which in Chinese keyword search, it takes some more time for the subjects to input Chinese characters.

- In office2, the Chinese version required more time than the English version except the second task—sub-category browsing. The Chinese keyword search required more time than English even though the task for English keyword search is more complicated than the Chinese task. Two of the results are significant.

In summary, for the Food Market application, tasks in Chinese were easier for the subjects except the keyword search while for the Office Supplies application, tasks in Chinese required more time than the English version. That is, less time was spent with the Chinese interface for the Food Market application while less time was spent with the English interface for the Office Supplies application. Thus we accept of first hypothesis (H1) that use performance in different language environments is related to the product characteristics.

We did not find any significant differences in task performance based on the type of interface (category, sub-category, or keyword) that was used for searching for product information. Hence we reject the second hypothesis (H2) that user performance in different language environments is related to search methods used. However, several interesting characteristics about user performance were observed. For example, users preferred to use category or sub-category methods to keyword search methods. A simple explanation for this could be that keyword search required more effort in terms of thinking of and typing a certain keyword that would be applicable to describe product characteristics.

Item Analysis

An important feature of user performance has to do with how users understand (or misunderstand) the information about products provided in an electronic catalog. We found through our experiments that there is a difference in the level of understanding for multilingual users when it came to product characteristics. For example, in response to the task of selecting a list of spices for cooking “seafood”, users exhibited a dramatic variation in their choice of the list of spices when the interface was in English that when the interface was in Chinese. This is illustrated in Figure 6, where Group1 and Group2 used the English and the Chinese language interfaces respectively. The numbers in the third column indicates the number of subjects that selected that particular item. As can be seen from Figure 6, the two groups selected different sets of spices for the same food item. For example, seven people using the Chinese language interface selected “hoisin sauce” as a spice required for cooking seafood, while none that used the English language interface selected it.

Figure 6: Results of Item Analysis

Result of Group2:			Result of Group1		
海鲜酱	hoisin sauce	7	pure pepper	纯辣椒	6
冠珍海鲜酱	koen chun hoisin sauce	5	rice cooking wine	香米料酒	6
上等麻油	sesame oil	4	spicy garlic sauce	鱼香酱	5
姜葱蓉	minced ginger & shallot	3	diluted red vinegar	大红浙醋	3
陈年酱油	aged soy sauce	3	grain vinegar	龙景米醋	3
辣豆瓣酱	chili bean sauce	3	pure sugar	精糖	3
龙景米醋	grain vinegar	3	chili bean sauce	辣豆瓣酱	2
香米料酒	rice cooking wine	3	kimlan soy sauce	金兰酱油	2
纯辣椒	pure pepper	2	minced garlic sauce	川霸王蒜蓉酱	2
大红浙醋	diluted red vinegar	2	minced ginger & shallot	姜葱蓉	2
金兰酱油	kimlan soy sauce	2	sesame sauce	芝麻酱	2
味精	ve-tsin	2	aged soy sauce	陈年酱油	1
五香粉	five spice powder	2	black pepper sauce	黑椒汁	1
鱼香酱	spicy garlic sauce	2	five spice powder	五香粉	1
川霸王蒜蓉酱	minced garlic sauce	1	satey paste	沙茶酱	1
黑椒汁	black pepper sauce	1			
精糖	pure sugar	1			
宫爆酱	Kung pao sauce	1			
沙茶酱	satey paste	1			

It may be possible to generalize this result by testing whether language has an impact on understanding and performing complex tasks. For example, it is possible that users will exhibit no language preference if the task is structured but may have significant language preference for unstructured tasks. Testing language preference over task/information complexity can provide insight into knowledge management in multilingual environments.

We performed the path analysis as described in the Analysis of Results section; however, we did not find any significant differences in performance for either the application domain or the type of interface used.

The experiment described above has several limitations for generalizing it to broader domains. The first limitation is the use of students as the subjects. The students cannot be viewed as the ideal sample of the real users. In our experiment, we tried to reduce this effect by selecting students from different departments. To some extent, students with different backgrounds can be viewed as the sample of real users.

The second limitation is the small sample size. We involved 20 students in our experiment. They were divided into two groups, ten in each. Thus the experimental results may suffer from small sample size effect. One of the effects is that some of results are not statistically significant although the difference between the two groups is significant. For example, the G1 (AVG) in Office1 is 244.3 and the corresponding G2 (AVG) is 182.2, which are obviously different, but not significant. A larger sample size with a smaller standard deviation may resolve this limitation.

Another limitation may be that there were no real purchases made in our experiment. The fact that the subjects did not spend real money for purchasing products may have had some influences on user performance. However, we assume that this factor would have influenced both groups identically.

RESEARCH FINDINGS

From the experimental results described above, we can surmise several implications for future research and practice in global electronic commerce. Firstly, users clearly indicated that they preferred to use their ethnic language (Chinese, in this case) when searching for ethnic products because it is difficult to translate them into another language such as English. Hence although the *unified content model* (Doherty 1999) is easier to implement in a global electronic commerce system, it may not be a good strategy, from a user interface standpoint, for all product categories.

Another closely related issue is the level of understanding of the information presented for bilingual users. The experimental results indicate that the subjects performed tasks equally well in both Chinese and English when dealing with structured information. However, when dealing with unstructured information, there was a significant improvement in their performance when the language of interface was Chinese as opposed to English. While it will take more research to establish a relationship between language preference and information complexity, such research can render significant insights into the design of multilingual interfaces for global electronic commerce.

In multilingual systems that depend on word-for-word translation from one language to another, a lot of information may be lost during the translation process. For example, it is hard to tell the differences between “table” and “desk” in Chinese. In these cases, images may be helpful to present product information. It is also worthwhile to study the effect of multimedia information on user performance and satisfaction in global electronic commerce.

CONCLUSION

In this paper we have described the design and implementation of an electronic catalog and experimental results for testing language preferences for users based on the task and product characteristics. The catalog has been implemented in English and Chinese and two applications have been developed: *office supplies* and *food market*. For each application, there are three methods that a subject can use to search for a certain item: *browsing*, *directed search*, and *exact match* using keywords. During experimentation, the performance of each subject is measured in terms of time required, access path used, and the items selected in performing the task. Through these experimentations we have established a linkage between

language preferences and task and product characteristics in using electronic catalogs.

Future plans include conducting larger experiments as laid out in the paper and analyzing the results to establish or reject such a linkage. The electronic catalog will be extended to a multilingual interface. Detailed testing will be performed for testing content management for *custom* and *hybrid content* models (Doherty 1999).

REFERENCES

- Adam, N. R., Dogramaci, O., Gangopadhyay, A., and Yesha, Y. (1998). *Electronic Commerce: Business, Technical, and Legal Issues*, Prentice-Hall.
- Broin, U. (1999). "International Aspects of User Interface Design," *MultiLingual Computing & Technology*, 10 (3), 50-54.
- Computer Economics. "English will Dominate Web for Only Three More Years," www.computereconomics.com/new4/pr/pr990610.html.
- Doherty, W. (1999). "Creating Multilingual Web Sites," *MultiLingual Computing & Technology*, 10 (3), 34-37.
- Malhotra, Yogesh (1997). "Knowledge Management in Inquiring Organizations," in the *Proceedings of 3rd Americas Conference on Information Systems* (Philosophy of Information Systems Mini-track), Indianapolis, IN, August 15-17, 293-295.
- Malhotra, Yogesh (1998). TOOLS@WORK: Deciphering the Knowledge Management Hype, *Journal for Quality & Participation* special issue on Learning and Information Management, 21(4), 58-60.
- Nirenburg, S. (1992). Text Planning with Opportunistic Control. *Machine Translation Special Issue on Natural Language Generation*, R. Kittredge (ed.), 7(1-2), 99-124.
- Onyshkevych, B. and S. Nirenburg. (1995). A Lexicon for Knowledge-Based MT. *Machine Translation*, 10(1-2).
- Segev, A., Wan, D., and Beam, C. (1995). "Designing Electronic Catalogs for Business Value: Results of the CommerceNet Pilot," technical report, The Fisher Center for Management and Information Technology, Haas School of Business, University of California, Berkeley, October.
- Sheremetyeva, S.O. and S. Nirenburg (1996). Empirical Modeling in Computational Morphology. *Nauchno-Technicheskaja Informacija* (Scientific and Technological Information), Series 2, Issue 7.

ENDNOTE

¹By individual preference we mean that the choice of the spices may vary from one user to another.

Chapter 3

End Users as Expert System Developers?

Christian Wagner
City University of Hong Kong

Knowledge is receiving recognition as a strategic force in organizations. Correspondingly, one form of knowledge capture and maintenance organizations are tempted to use is expert system design by end users. The article discusses difficulties associated with end user development, both in terms of design quality and knowledge content. An analysis of 25 expert systems written by non-professional developers reveals significant quality and size limitations that indicate limited feasibility of end user expert system development. Furthermore, the lack of design quality may not be easily compensated for by a “knowledge advantage” of the end users, as end users may have a performance advantage in using their knowledge, but not in “knowing” it.

BACKGROUND

Growing Need for Knowledge-Based Systems

Knowledge is gaining widespread attention as a strategic tool for the competitiveness of firms, both in the management literature and the popular press, e.g., Nonaka and Takeuchi (1995) and “In Praise of Knowledge” (*Economist*, May 27, 1995, p. 20). However, knowledge creation and management is not simply the capture and storage of information. It also requires the storage and processing of associations (rules) through which meaning can be derived from the

information. And, to follow the argument of Nonaka and Takeuchi, knowledge should be stored in explicit, observable form.

Presently, the pre-eminent information technology vehicle for the explicit representation of knowledge in managerial applications is the expert system (or knowledge-based system). Thus, knowledge explication would require massive expert system development. Given the existing demands placed on the systems development function, the implication is that at least part of the expert system development effort will have to be completed by end users. This creates a dilemma. On one hand, the bottleneck in knowledge acquisition is well acknowledged (Holsapple and Raj, 1994). On the other hand, one of the truisms of expert systems development is that the domain expert should not be his or her own knowledge engineers (Hayes-Roth et al., 1983), a heuristic based on Johnson's *paradox of expertise* (Johnson, 1983).

Despite common wisdom condemning end user expert system development, there are numerous accounts of successes with this development approach. For example, DuPont (Feigenbaum et al., 1988; McNurlin, 1987) reported positive experiences with end user created solutions 10 years ago when development tools were more primitive than today. So did Eastman Kodak (Huntington, 1989), and so did the U.S. Navy (Griesser and Tubalkain, 1992). The number of applications ranged from a few tens (U.S. Navy) to more than 1,000 (DuPont), the size per application from tens of rules to several hundred. Lubrizol has users participate in expert system development by letting them build the underlying decision tables (*I/S Analyzer*, 34(3), 1995, pp. 12-15). In addition, several of today's development "languages" have become much more user friendly (e.g., visual design of trees as in Exsys' Rulebook software) and are now targeting end user developers rather than professional developers. World wide sales for these desktop AI tools amounted to US\$4 million in 1995, with annual growth rates exceeding \$500k ever since 1991 (*Intelligent Software Strategies*, XII(2), 1996, p. 4). These trends are paralleled by numerous university curricula offering courses on expert systems to their business (!) students, who are likely not going to be professional developers.

Clearly, end user expert systems development enjoys at least a successful niche presence. Yet a recent survey on end user computing sophistication (Blili et al., 1996) did not even consider to ask participants about the use of fifth generation expert system tools. Hence it seems that most companies do not even consider this approach, while those who do are successful (or at least claim success). Will therefore those companies who embrace user developed expert systems become knowledge-based firms, while the many others will not? And, what are the limits to end user developed systems? Accounts of successes, such as those of DuPont,

Kodak, or the Navy, are not sufficient to help us answer this question, as they are always presented at the macro level (company totals for the number of systems developed, number of rules per system, average pay-back). They report little about the difficulties end user developers face in actually creating their systems.

Special Problems of End User Developed Expert Systems

In a recent study, Panko (1998) addressed the issue of spreadsheet errors and their potential impact on organizations. He reports a picture of unacceptably high error rates and significant bottom-line impact related to these errors. If a similar pattern emerged in expert system development by end users, it would already be concerning. However, expert system development adds several additional elements of complexity, compared to spreadsheet development, which can potentially result in even higher error rates, namely:

- Expert systems carry out qualitative (logic based) reasoning using (frequently) non-numeric symbols whose semantics can be defined by user rules. Spreadsheets are largely quantitative with well-established semantics.
- Expert system developers will usually have little training in the “language” of expert systems, which might be a derivative of first order logic, or a similar logic-oriented language. In contrast, spreadsheet developers can rely on many years of training and expertise in the common “language” of spreadsheets, arithmetic.
- Most expert system shells separate development from use. That is, the knowledge base will be developed in an editor or other development environment (with limited error checking). After completion of the knowledge base, it will then be executed. This separation of development and use makes verification and correction of errors significantly more tedious, if not difficult than in spreadsheets, which are more WYSIWYG in nature and which provide instantaneous feedback on the correctness of a formula, or on the impact of a number change.
- Rule-based expert systems frequently adopt a backward reasoning approach whose logic is not at all obvious to inexperienced developers, or to developers with experience in traditional programming languages.

Taken together, these characteristics add extra complexity which affects knowledge extraction and formalization (extraction and formalization of “rules” or such), representation (especially in a backward reasoning environment), validation (in the absence of WYSIWYG and instantaneous confirmation), and learning/usage (logic with user definable semantics versus algebra). Consequently, higher error levels are to be expected than for more typical end user applications such as spreadsheets, and special attention needs to be drawn to key sources of errors.

Overview of the Study

Given the apparent difficulty of end user expert system development on one hand, and the increased demand for end user developed expert systems on the other hand, the decision was made to investigate the feasibility of this form of knowledge base creation. Two questions were addressed. First, *are end users able to develop expert systems that are structurally of sufficient quality* to represent the knowledge they are supposed to (knowledge formalization and representation), and secondly, *are experts capable of expressing their reasoning* sufficiently to create valuable knowledge bases (knowledge acquisition)?

The first question was investigated by assessing the quality and size of end user developed knowledge bases. The purpose was to determine the limits for size and reliability of end user created solutions and thus to assess how far end user expert system development can go. The second question was addressed through the review of research in the area of tacit and implicit reasoning. Its purpose was to find evidence for whether end-user developers can develop expert systems that convey useful business knowledge, regardless of design quality.

The rest of the article is organized as follows. First, an empirical analysis of end user developed applications is introduced, together with metrics for expert system quality. This is followed by a more detailed analysis of the problems encountered in end user development. The purpose of this section is to illustrate to the reader which aspects of the development process create difficulties for the inexperienced developer. Thereafter, the issue of knowledge explication is discussed. The last section extrapolates the findings to determine the limits of end user developed expert systems and discusses alternative approaches to capture the valuable knowledge of end users.

DEVELOPMENT BY END USERS: EMPIRICAL ANALYSIS OF KNOWLEDGE BASE DESIGNS

End User Development of Expert Systems

End user developers in our consideration are professionals or managers whose primary function is not information systems development. These individuals will usually select a lower-end development environment (e.g., M.4, Exsys, or similar), based on simplicity and cost, and will choose application areas according to their own knowledge and interests. To study this form of development and the problems associated with it, 25 expert system development projects were analysed. The systems were written as individual in-class projects by MBA students (84% part-time students with full-time professional or managerial jobs). Average work

experience exceeded five years. The course was a Year 2 (2-year program) elective in management support systems for students who had taken at least one prior course in information systems, and some course-based, hands-on software development experience. Course participants chose problem areas to develop systems based on their individual interests, but according to problem feasibility criteria, as outlined in Waterman (1986). To enable them to develop expert systems, course participants received approximately 15 hours of instruction, covering both methodology (e.g., development life cycle) and implementation issues. The instruction duration was similar to that of courses offered in industry (usually 2 to 3 days, as for instance at Knowledge Garden, URL <http://www.kgarden.com>). Despite these hours of instruction, virtually all participants were novices with respect to the task.

The project counted for 25% of the participants' course grade, thus offering a strong incentive to do well. Participants also had the ability to do well, as they received clearly formulated evaluation guidelines, including a diagnostic sheet. Participants also submitted milestone reports and received feedback on their milestone reports with respect to fulfillment of the evaluation criteria.

The programming environment was Cimflex-Teknowledge's M.4, successor to one of the early commercial shells, M.1. Its inference engine permits both rule-based and frame-based reasoning (through class definitions), as well as probabilistic reasoning. M.4 operates under MS-Windows 95 as well as under MS-DOS.

Metrics for Quality Measurement

Measuring software quality has long been recognized as a complex issue. In the software engineering literature therefore, quality is often very broadly defined as conformance to specifications (Crosby, 1984), and is operationalized through attributes such as those identified by McCall et al. (1977). These guidelines are useful for professional software development, but much less so for end user development, where specifications may only exist (vaguely) in the end user developer's head.

This study therefore adopted an approach to measure three criteria which are applicable to a variety of development projects, whether systems professional or end user. The first criterion, *completeness*, is related to the content of the software to be created. The second criterion, *decomposition*, reflects good design thinking, but also strongly impacts maintainability of the knowledge base. The last criterion, *probabilistic reasoning*, reflects a key aspect of expert systems, namely their heuristic nature. As is the practice in measurement of software quality, quality criteria (at least two of them) were measured through different operationalizations (metrics).

The first quality criterion was *completeness*. Unfortunately, it is almost impossible for end user developed applications to assert whether they are “complete”, or what their level of completeness is. Consequently, two measures were chosen. One was the absolute size of the knowledge base measured through number of rules. Size can be considered a surrogate for richness of the knowledge base, as it determines the number of conditions the knowledge base can handle. This however is true only for two knowledge bases of the same level of decomposition (discussed below). The second operationalization was a measure of the presence of “holes” in the knowledge base, a determination of knowledge base incompleteness. A knowledge base hole exists, when there are conditions for which the knowledge base has no response, even though they belong to its domain. For example, the knowledge base may offer investment advice for when the stock market goes down or goes up, but not advice for a stable market. If common wisdom (Johnson, 1983; Hayes-Roth et al., 1983) were to prevail, this quality criterion should be compromised by end user developed systems.

The second criterion was *decomposition*. Decomposition is widely accepted as a design principle in software development (Myers, 1978; Parnas, 1972), and in database design (Bernstein, 1976; Date, 1990; Ullman, 1980), and now also in expert system development (see Coenen and Bench-Capon, 1993; Hicks, 1994-95). Design rules, such as modular design or use of structured programming concepts, or database normalization (Date, 1990) are illustrations of this concern. Fundamentally, designers are advised to design systems so that components that belong together are associated with each other (cohesion), and that those that do not belong together are separated (decoupling). Yet while these concepts are well accepted in professional systems development, they might not be at all intuitive to end user developers. To measure decomposition, three metrics were used, namely (largest) number of premises in a rule, levels reasoning within the knowledge base, and the use of OR connectors, all of which will be described in more detail in the next section. Although this criterion might be perceived as less important for user developed applications, violations of rules of decomposition will limit growth and hamper maintenance, and therefore limit the use of the developed systems as an organizational asset. Stockdale and Wood (1992) also identified the issue of knowledge base structuring as an area of weakness for end user developed systems.

The third criterion was the use of *probabilistic reasoning* within the knowledge base. Part of the appeal of expert systems is their ability to incorporate an aspect of expert reasoning which is reflected in approximate reasoning, use of heuristics, and pursuit of multiple paths. As professionals in their field of expertise, the individuals participating in this study were aware of non-deterministic concepts. In fact, they had received formal instruction about heuristic reasoning and the

mechanisms of probabilistic reasoning within expert systems. They therefore had “motive and means”. Would they then follow through, and in fact implement this form of reasoning in their systems?

Were useful criteria left out in this analysis of the quality of user developed systems? Likely. For once, the study did not measure the content quality of each knowledge base. Thus, a knowledge base could “score highly” but at the same time be simple in its advice capability, if the developer chose such a direction. Knowledge bases were not allowed to be trivial, but there was clearly a range of content quality. However, it was impossible to prescribe to study participants which area to build a system for, because the purpose of the exercise was to let them build a system for an area they were highly familiar with. Hence there was no uniformity in the knowledge base content and no well justifiable way to assess content quality *per se* (the issue of knowledge content therefore will be addressed in this article in more general terms based on research into people’s ability to express their knowledge). Also left out were typical measures such as bugs that would not allow a knowledge base to execute. A minimum qualification for all systems was that they were executable. Overall then, the quality measures used in this study covered—although not completely—the two defect areas associated with the knowledge based view of software maintenance (Coenen and Bench-Capon, 1993), *content* and *structure*. Together with the probabilistic reasoning criterion, they were thus expected to provide a multi-faceted insight into the quality of user developed systems.

Subject Performance

All 25 projects were evaluated according to the six measurement criteria. The results are listed in Table 1. Table 1 shows an overall wide range of values for the six metrics, representing different level of sophistication and development effort. In total, quality concerns were detected in 80% of the projects. Eight of 25 projects had missing rules, five gave strong indications of poor decomposition, and 14 lacked probabilistic reasoning. At the same time, knowledge bases were not insignificant in size, with an average of 78 rules each. Looking at the table one might wonder what levels of performance are acceptable. For some criteria, the answers are simple. Holes, for instance, are not acceptable at all. For other criteria there are no such hard-and-fast rules. Hence the next section will discuss good and less good performance parameters and will highlight particular design weaknesses.

END USER DEVELOPMENT DIFFICULTIES

Completeness Problem: Holes in the Knowledge Base

Although the systems developed as part of the exercise were of considerable size (average 78 rules), almost 1/3 had missing rules. As a result, such a system

Table 1: Project Summary Data

Project No.	Application	Max. Number of Premises	Number of Rules	Number of Levels	Use of OR	Prob. Reasoning? in the KB?	Holes
1	Help desk	5	47	2	No	No	Yes
2	Help desk	4	57	3	No	No	No
3	Help desk	5	24	2	No	No	Yes
4	Help desk	3	39	3	Yes	No	No
5	Employee evaluation	9	26	2	No	No	No
6	A/V equipment selection	5	108	2	No	No	Yes
7	Equipment selection	5	44	3	Yes	Yes	Yes
8	Dining out selection guide	4	75	3	No	No	No
9	Employee morale advisor	4	86	4	No	No	Yes
10	Blackjack advisor	4	128	4	Yes	Yes	No
11	Travel advisor	4	83	3	Yes	Yes	No
12	Financial aid advisor	5	87	4	Yes	Yes	No
13	Software explanation	3	59	4	No	No	No
14	Order audit	4	35	3	Yes	Yes	No
15	Insurance profiling	3	95	3	No	No	No
16	NFL wager advisor	8	37	2	Yes	Yes	Yes
17	Space system architecture planner	3	175	5	No	Yes	No
18	Risk assessment	3	32	4	No	No	No
19	Computer configuration	3	244	4	No	No	No
20	General problem solver	6	109	3	Yes	Yes	No
21	Clothing "catalog"						
	(intelligent DB)	5	155	2	No	No	Yes
22	SBA loan eligibility advisor	7	30	2	No	Yes	No
23	Medical advisor	2	51	3	No	Yes	No
24	Help desk	4	33	3	No	Yes	Yes
25	WWW advisor	19	91	3	Yes	No	No
<hr/>							
		Average No. of Premises	Average No. of Rules	Average No. of Levels	Total with OR	Total with Probs.	Total with Holes
		5.1	78.0	3.0	7	11	8

frequently answers “I don’t know” (or the equivalent in “expert system language”) to an inquiry. This occurs, because the system does not contain a rule that governs the combination of conditions specified by the user. Although likely to be expected from an unfinished prototype, this is not necessary. Holes are frequently due to an oversight, where part of a range of values is not covered. This is an easy mistake for end users who frequently do not use knowledge formalization techniques, such as representation in decision tables. Holes become even more likely and more difficult to detect when the system is not well decomposed and therefore every rule contains many premises (see below). Other holes exist because the novice developer assumes that specific conditions or combinations of conditions just cannot occur and therefore can be left out.

Decomposition Problem: Flat Knowledge Base

A “flat” knowledge base, like a flat organization, has few layers. The number of layers or levels in a knowledge base can be determined as follows. The top level goal is at Level 0. Rules that will satisfy the goal are Level 1, rules that are called by Level 1 rules become Level 2 rules, and so on. At one point, there are no further calls to rules (the expert system then asks the user or consults a database). The highest level of rules then describes the depth of the knowledge base. The minimum level of depth for a knowledge base is 1 (it can be 0, if the knowledge base contains no rules at all, but a knowledge base without rules does not really qualify as such). While flatness is desirable in modern organizations, it is not necessarily desirable in knowledge bases. Especially when the knowledge base gets larger, we should expect more than one or two levels of reasoning. In fact, none of the projects listed in Table 1 contains only one level. Nevertheless, an obvious example of a flat knowledge base is for instance Project No. 21 with 155 rules and only two reasoning levels. A flat knowledge base is the result of little decomposition of the application problem. Figure 1 demonstrates differences of depths and corresponding rules by means of a credit card application problem. (Note that for ease of reading, the rule syntax in the example does not follow M.4 syntax entirely, especially with respect to capitalization and arithmetic expressions).

In the example, the same knowledge is represented by one “flat” rule (somewhat of a misnomer, since it is not the rule that is flat, but the knowledge base it belongs to) versus four “deep” rules. The “flat” rule has six premises, while the knowledge base it is part of contains only one level, compared with two levels for the “deep” rule. At first, a comparison between the two representations appears to be in favor of the flat one. While it is a little longer (6 premises), one “flat” rule replaces four “deep” ones. If each premise is multi-valued, however, e.g., Home_Owner = Yes, Home_Owner = No, and has different corresponding outcomes, the flat knowledge base becomes quickly less favorable. While rules remain long, they also become plentiful.

Figure 1: Flat versus Deep Knowledge Base

Rule Flat-1:	
IF	Income > 3000
AND	Income > 3 * Debt_Service
AND	Home_Owner = Yes
AND	Years_at_Residence > 2
AND	Job = Yes
AND	Years_at_Job > 2
THEN	Approval = Yes CF 100.
Rule Deep-1:	
IF	Finances = Stable
AND	Home = Stable
AND	Employment = Stable
THEN	Approval = Yes CF 100.
Rule Deep-Finance-1:	
IF	Income > 3000
AND	Income > 3 * Debt_Service
THEN	Finances = Stable.
Rule Deep-Home-1:	
IF	Home_Owner = Yes
AND	Years_at_Residence > 2
THEN	Home = Stable.
Rule Deep-Employment-1:	
IF	Job = Yes
AND	Years_at_Job > 2
THEN	Employment = Stable.

Rule length is a disadvantage of flat design. Although any two rules may only differ in a single attribute value (e.g., Home_Owner = Yes vs. Home_Owner = No), each rule has to be written and maintained. Longer rules are unfortunately also more difficult to verify than shorter ones.

The flat design's second weakness, its need for more rules, may be illustrated quantitatively through the following example. If we consider the scenario in Figure 1 and assume that there are two values for each premise, then the results are as follows:

- Flat design (six premises) results in 2^6 rules = 64 rules.
- Deep design (one rule with three premises, three rules with 2 premises each) results in $2^3 + 3 * 2^2$ rules, for a total of 20 rules.

Of course, the larger the number of significant values (e.g., more than two significant values of income or years-with-employer), the more pronounced the differences become. Again, more rules typically mean more development, maintenance, and verification effort. This is where holes easily appear in the knowledge base. When the number of rules becomes large (i.e., 64), one of them is easily forgotten. In the deep design, fewer rules have to be written. Furthermore, much fewer rules are written at each level, which simplifies verification even more.

A third disadvantage is the lack of intermediate results. This disadvantage is based on a feature of expert systems which lets them remember intermediate reasoning results of an inference, regardless of the final outcome of the inference. The intermediate results can then be used for other inferences during the same session. Unfortunately, the flatter the knowledge base, the smaller is the number of intermediate results. In the Figure 1 scenario for instance, the flat knowledge base reaches only one type of conclusion, namely whether to approve. The deep knowledge base, by comparison, also concludes the financial, home, and employment situation. Although we should not expect much performance (speed) improvement from them, intermediate results are potentially useful in explaining expert system recommendations to a user. For example, success of rule Deep-Finance-1 may prompt the explanation “the applicant’s financial situation meets the approval criteria.”

Among the projects listed in Table 1, there are several knowledge bases which are in apparent need of further decomposition. Project No. 6 for instance consists of 108 rules, yet has only two levels. Project No. 25 possesses three levels, but also rules with too many premises (up to 19 in one rule). In contrast, Project No. 23 is highly decomposed. It has no rule with more than two premises, contains 51 rules, organized in three levels of reasoning.

Given the disadvantages associated with flat knowledge bases, why would designers actually create them? Novice developers report that flat knowledge bases are easy to design. The developer sets up a “rule template” and then produces variants, often simply through “cut-and-paste”. This procedure can initially quickly produce a small knowledge base that deals with some of the relevant case scenarios. It is only later that the developer realizes how difficult it is to complete the knowledge base and to detect any existing errors in it. In contrast, creating a deep knowledge base requires more initial insight and planning. The developer has to understand how to decompose the knowledge of the domain and then implement the knowledge in decomposed form. This, of course requires more understanding of design principles, and is a skill that end user developers will rarely develop.

Decomposition Problem: Rule Coupling through OR

Some designers may consider the use of OR within knowledge base rules an efficient mechanism to reduce the overall number of rules. And it can be. After all, the use of OR allows us to combine two sets of premises that result in the same conclusion, as illustrated in Figure 2. In fact, even the M.4 manual exemplifies the use of OR in knowledge bases. In our set, seven out of 25 projects used OR statements in rules. Yet before two sets of premises are combined, we need to ask whether they actually belong together. Are they related in meaning other than their simply inducing the same conclusion? The two premises in rule Approval-1 in

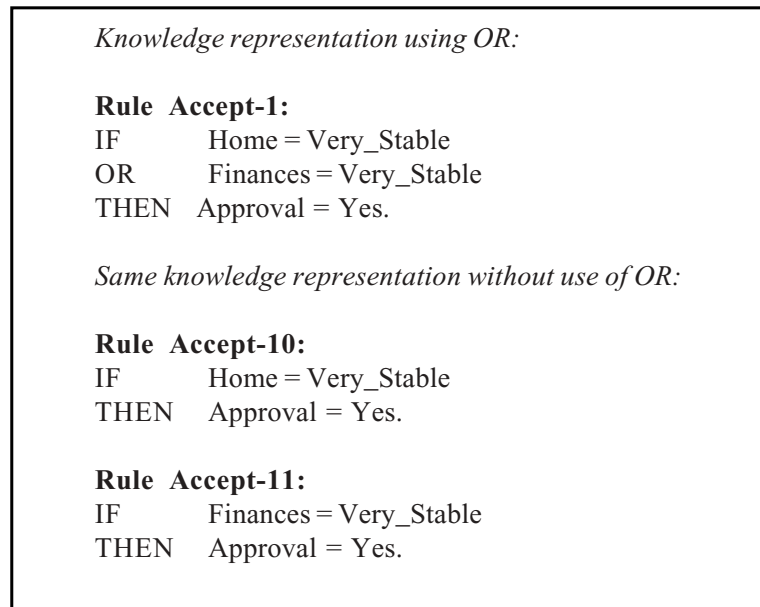
Figure 2: Use of Logical OR Statement

Figure 2, for instance, have little in common, they are not related. Premises should be considered related, if they refer to the same condition and differ only in the value for that decision, e.g., IF Home = Very_Stable OR Home = Stable THEN ...). Again, concepts of programming and database design advise us not to combine unrelated elements. In programming, this would be considered coupling (7). When rule premises are coupled, as in rule Accept-1 in Figure 2, the second, independent premise is much harder to detect during knowledge base verification or modification. For example, when all statements relating to Finances need to be changed, the “hidden” second premise in Accept-1 may be overlooked. Alternatively, if the developer decides that the first premise does not apply anymore, he or she may accidentally remove the entire rule, thus eliminating also the second premise.

A representation without OR has further advantages if combined with probabilistic reasoning. When premises are separated, one can attach a certainty factor to the conclusion of each separated rule and increase the cumulative confidence factor for a conclusion, as more and more evidence is gathered. The details of this approach are discussed further in the next sub-section.

When developers use OR, it is often not to shorten the knowledge base, but because inexperienced developers draw analogies to traditional programming where both AND and OR are common. If AND is used in expert systems, then why not also OR? It requires some experience for novice developers to realize that writing a new rule (with the same conclusion) is identical to attaching a premise with an OR, so that the use of OR is in fact unnecessary. Unfortunately, when rules have (too) many premises, the attachment of an extra condition with an OR becomes seemingly more efficient than writing an extra rule.

Lack of Probabilistic Reasoning

End user developers often do not seem to realize that the knowledge they represent is probabilistic instead of “hard and fast”. As a result, rules that should be described using confidence factors are stated as certain (see the example in Figure 5, rule Approval-100). For example, the majority of projects ($\frac{14}{25}$) listed in Table 1 was programmed in deterministic form, even after multiple reminders to consider probabilistic reasoning. Two reasons were given for this type of representation. First, based on their verbal comments, developers apparently truly believed their tasks were non-probabilistic, even when they were obviously not. This, by itself, is an interesting issue. Experts may recognize that the rules they know do not work all the time, and are comfortable in applying “back-up” rules, if the most likely ones do not work. At the same time, they seem to feel uncomfortable in operationalizing this method of reasoning in form of most-likely rules with high probability values and less-likely rules with low probabilities. Second, being novices, the developers considered the introduction of meaningful confidence factors too difficult. They simply found it difficult to attach a specific probability (or certainty) value to a rule. It should be noted here that even experienced developers may find the assignment of factors representing the probabilistic nature of the experts’ knowledge difficult. However, they typically see the problem as one of determining the most appropriate probability values (or certainty factors) so that final conclusions are presented with meaningful probability numbers.

End-User Expertise: Knowing More than They Can Tell?

One potential advantage of end user developed knowledge bases is that end users are the source of domain expertise. Thus, instead of explaining their knowledge to a knowledge engineer, end user developers can save the intermediate step of trying to “educate” another person who then implements the knowledge. The more direct process of end user development would therefore hopefully create a richer and more accurate knowledge base. Consequently, weaknesses in structural design would be compensated for by strength in knowledge and value of that knowledge. The question then is, how able are end users to elicit their own knowledge?

Waterman (1986) provides an insightful example of knowledge acquisition, which may illustrate the issue. In Waterman’s example, an insurance claims expert is interviewed by a knowledge engineer to elicit and formalize the expert’s knowledge. The expert, after reading the case detail, quickly gives an estimate of financial liability without explanation. Probed by the knowledge engineer during the subsequent discussion, the expert explains his assessment, describing algorithms and heuristics used, as well as assumptions made. In the end, the expert’s lengthy

computations result in a financial liability figure almost identical to the value determined prior, without explanation (difference of about 1%).

The example suggests at least two points. First, the expert may use different reasoning mechanisms for the same problem, a fast and efficient data or case-based reasoning format to solve the problem, and a formal, rule-based reasoning format to generate explanations. After all, if the same reasoning were used, results would be identical. Second, experts are expert in what they do, not necessarily in explaining (which is more the expertise area of teachers and university faculty). An expert may not be able to provide explanations for all of his or her reasoning. While being able to solve the problem, he or she will not be able to explain the necessary rules or heuristics to do so.

Evidence for the existence of unexplainable “tacit” knowledge has mounted in recent years. The existence of tacit knowledge was put forward a long time ago, however, by Polanyi (1966). Polanyi’s views of tacit knowledge can be stated as follows:

Tacit knowledge is a set of facts and rules, of which only the resulting (i.e., implied) facts are observable by the knowledge owner. The resulting component, called *Term 2*, is considered *specifiably known*. The unobservable component (*Term 1*) is considered *tacitly known*.

For example, a loan officer might review the facts of a loan application and reject the application. Asked “Why did you reject it?” she cannot give a meaningful explanation (“it did not warrant acceptance”), because she is not aware why she rejected it.

Term 2 (specifiably known): loan rejection, (fact).

Term 1 (tacitly known): facts, such as monthly income, monthly debt service, rule stating “reject an application if monthly debt service exceeds 20% of monthly income.”

Evidence for Tacit Knowledge and Implicit Reasoning

Empirical evidence for the phenomenon “knowing more than being able to tell” has emerged from multiple task areas. For example, in several experiments, Lewicky et al. (1987, 1988) found that subjects were able to improve their performance in cognitive tasks over thousands of trials, but were unable to explain the rule that governed their performance, even when offered rewards for explaining, amounting to \$100. In another study, Reber and Lewis (1977) found that through task performance subjects became better at the task as well as better in explaining their performance (explication), however that their performance increase surpassed their improvement in explaining the performance. In yet another study, Mathews et al. (1989) demonstrated empirically that when a subject group, while learning a task, explained its knowledge to a control group, the control group

performed at about half the level of accuracy of the first group. These examples point to an important shortcoming of the expert (our end user) working alone, and stress the need for a knowledge engineer not simply as a knowledge implementer, but as a knowledge creator who observes the expert's behavior and translates observations into rule (or similar) form.

An example may illustrate this issue further. Let us assume, a person is asked how to add two numbers. The person's answer could be, "for as long as the first number is greater than zero, decrement it by one and increment the second number by one. Once this process of counting down and counting up is finished, the second number will reflect the total." Let us also assume, that same person is asked to calculate the result of $3 + 4$. He or she will likely reply "7", but will likely not apply the rule just described, but instead simply recall the result from memory. Being able to recall results from memory with accuracy and consistency will suggest the use of a rule (such as that given earlier). Yet even if unable to provide a rule for how to add up numbers, many people will be able to carry out arithmetic correctly through recall from memory. Similar principles are at work when an expert, as in Waterman's example, quickly gives an estimate that seemingly requires a complex formula to determine it. The expert recalls a fitting case from memory (an "anchor") and adjusts it to reflect the differences between the recalled solution and the characteristics of the task ("adjustment"). Compare Tversky and Kahneman (1974) for the value and perils of this reasoning heuristic.

The discrepancy between the ability to complete the task and "knowing the rule" is supported through the findings of Reber et al. (1980). This study found that "knowing the rule" hindered subjects' task performance compared to simple pattern matching performance. In the study, subjects learned the problem task experientially. Having achieved a reasonable performance level, they were told the "rule" by which the pattern matching problem was solved. Yet being told the rule (which in many cases did not match with subjects' own understanding of the task) slowed their performance.

This and related results (Perruchet et al., 1990) suggests that people often simply "don't know" their own knowledge. More than being unable to explain, they simply have no complete understanding or a partially incorrect understanding of the underlying principles. This is different from the argument made by Johnson (1983) and many of the issues discussed by Gaines (1987). While to an outside observer it appears as a set of carefully crafted rules, expert knowledge might better be described as a fairly complete and consistent set of case-based solutions (some of which the experts can explain). Irton et al. (1990) support the argument in their analysis of the development of five expert systems. In one of the development cases, the experts frequently frustrated knowledge by providing highly specific cases, when the knowledge engineers were looking for general principles and rules.

In summary, then, the end user's domain knowledge might not be in a form that is immediately useful for implementation into an expert system. Even if an expert can describe a rule by which the reasoning is supposedly carried out, it may not at all describe the expert's reasoning mechanism. Hence, the end user's best knowledge, the knowledge that goes beyond routine problem solving and textbook knowledge, might only be discovered (!) through a knowledge elicitation and interpretation process which involves a skilled knowledge engineer as well.

IMPLICATIONS FOR KNOWLEDGE MANAGEMENT

The discussion so far has revealed shortcomings in end users' ability to construct quality knowledge and has suggested a further inability to express key problem solving knowledge as well. The implications of both these concerns are discussed below.

Limits to Knowledge Base Structural Quality

The projects analyzed here were of considerable size, yet they also exhibited a range of problem symptoms. Size is seemingly easier to generate than good design. Lack of probabilistic reasoning indicates that developers did not fully capture the essence of expert reasoning. Missing rules point to end user difficulties in the systematic validation of knowledge bases. Poor decomposition is evidence that developers ignored, willingly or unwillingly, knowledge base decomposition principles (thus creating future maintenance and enhancement headaches). Taken together, end user expert systems make trade-offs that qualify them as "throw-aways", systems that might work at the present, but lack the features that allow them to easily grow and be maintainable.

These quality concerns were encountered even though all individuals had been previously instructed in knowledge base decomposition, probabilistic reasoning, use of OR, and knowledge base verification. In addition, all projects had gone through one prior development iteration (participants submitted an early version for review and comments).

Limits to Knowledge Base Size

With knowledge base design and maintenance being time consuming tasks, there are obvious limits to the maximum size knowledge base a developer can create. If an end user spends about as much time per year on expert system development as the participants did in this exercise (about 50 hours on average), the resulting systems should also be of similar size. That size, around 80 rules on average, is very much in the range of systems developed at DuPont or Eastman

Kodak. And while it is considerable already, 80 rules are not enough to capture significant amounts of expertise. By comparison, American Express' Authorizer's Assistant contains about 850 rules, Coopers & Lybrand's Expertax about 3000 rules, DEC's XCON system about 10,000 rules. Also, the professional systems contain well-written rules, where one well-written rule may represent the same knowledge as several poorly written ones.

The above discussion does not imply that a larger knowledge base is automatically a better or more knowledge- rich knowledge base. This was demonstrated earlier in the discussion of flat vs. deep knowledge bases. By comparison, the developers of XCON, for instance, measure size and complexity in terms of the number of rules, but in addition consider the number of conditions per rule, the number of conclusions per rule, and the number of attributes per condition element (Barker and O'Connor, 1989). With respect to richness, XCON is also interesting because its size fluctuated up and down. Periods of increase in rule size were followed by a re-organization effort in which the knowledge base size shrank, although the contained knowledge did not. Yet, all else being equal (i.e., two knowledge bases with same depth), the system with more rules will contain more knowledge and will provide more reasoning capability.

Does Type of Domain Matter?

One potential source of poor designs might of course be related to the choice of problem domain or the type of system built. As pointed out for instance by Waterman (1986) and Mallach (1994), there are "good" expert system problems and less good ones. In order to steer subjects away from less good problems, they had to have their problem choice approved and had to formally argue for the problem's suitability, using Waterman's criteria. Project topics are listed in Table 1. While there is little data available to suggest any strong patterns, it appears that the least sophisticated designs coincided with the helpdesk system applications. The typical weaknesses in these systems were flat knowledge bases (long rules) with corresponding holes in the knowledge base (3 out of 5). None of the projects dealt with toy problems. The least "expert" system was probably Project 8 (Dining out selection guide), yet it had a lengthy knowledge base. Another project, Project 16 (NFL Wager advisor) whose subject area might suggest a "toy application" was based on the developer's complex wagering system, which he had developed and successfully used over many years.

Good Knowledge in Poorly Designed Knowledge Bases?

Does poor knowledge base design imply a poor knowledge base? Of course not. A poorly designed knowledge base with holes and other structural deficiencies may nevertheless capture a valuable portion of a company's business rules. Yet the

scenario of an end user designed structurally poor, but knowledge-rich, expert system is not all too likely. A weakly structured knowledge base will have difficulty in representing deep knowledge, which is knowledge with many layers of structure where the result of one layer becomes the input of another. This type of knowledge is, however, a reflection of true expertise and one of the targets of end user developed expert systems. Another, more suitable type of knowledge might be categorized as “clever” knowledge. This would be knowledge that is simple in structure, but only known to few people and therefore of high value. Examples might include “smokers buy more furniture”, “beware of Greeks bringing gifts”, or “the murderer always returns to the scene of the crime” (some of these rules are less widely known and less clever than others). If knowledge of this type can be captured and propagated throughout the organization, it can provide broad value.

And yet, all other factors being equal, even with this type of knowledge, a well-designed knowledge base will be more valuable than a poorly designed one, because its structural quality will lead to better knowledge quality. There will be fewer holes, leading to less frustration on the side of other users using the expert system. There will also be fewer chances that appropriate rules do not fire, due to design problems. Finally, the maintainability of a well-structured knowledge base will allow it to change and grow.

Need for Knowledge Capture through Alternate Means

The limits encountered in end user expert system development create a dilemma. On one hand, companies do not have the resources to use professional developers (knowledge engineers) to codify significant amounts of end user knowledge. On the other hand, end users, while possessing the knowledge are unable to codify it themselves in a meaningful manner, using the available tools. *Groupware* products, such as Lotus Notes are partly suitable to overcome the dilemma, as end users are able to record their knowledge in them using “plain text”. But, described in plain text, knowledge is usually ambiguous, incomplete, and cannot be easily processed. *Data mining* tools, such as Knowledge Seeker, which extract rules based on past data hold some promise, especially for numerical data, but are not particularly suitable for creating for instance complete diagnosis expert systems. They are predominantly used in marketing (see 12/96 *META Group Study of 125 Global 2000 Companies*), to identify a few key rules (not a consistent set of rules), such as “smokers buy more furniture”. Furthermore, data mining is at present not an end user task. Companies not only use high-end hardware, software, and database components, but also employ trained experts, to make sense of the results.

CONCLUSIONS

As more and more firms recognise that it is easier to collect large amounts of data than to make sense out of that data, they look for intelligent information technologies to support that activity. Creating intelligent systems on a large scale will, however, remain a difficult endeavour. All “famous” intelligent information systems have been the product of costly professional development efforts. And, as this study indicates, there is little chance that similar successes can be achieved through end user developed systems. End user development will be limited in content, quality, and size and will not scale up. Ten 80-rule expert systems do not contain the same knowledge as one 800-rule system, and the development of one large system requires significantly more effort, planning, and developer ability than that of many small ones (Gallagher, 1988). Companies that look for new ways to create and manage knowledge will have to search for alternative means to achieve this goal. Yet while some new technologies promise knowledge discovery (data mining) and knowledge management (groupware), even their application has obvious limits, perhaps capturing just “clever knowledge”. Nevertheless, the positive payback organizations such as Kodak or the Navy have received from larger numbers of smaller systems should not be forgotten in the decision whether to decentralize expert system development.

Interestingly enough, although user-developed decision support systems have become commonplace and are well supported through tools such as spreadsheets or visual databases, the same is certainly not true for knowledge-based system development. But without the tools to formalize and represent knowledge at large scale by those who own that knowledge, moving towards a knowledge-based organization will be difficult. Spreadsheets allowed end users to become decision support system developers. Which new technology will enable end users to become knowledge system developers?

A related issue concerns business school curricula. With present technology, training students to become end user developers of expert systems appears infeasible for the majority of students. And yet the same development exercises may be helpful in educating those students to understand the nature of knowledge, how to extract it, and how to formalize it, even if the working prototype leaves great room for improvement. At the same time, those organizations that followed an end user expert system development approach spent much time and resources on training end users on all aspects of development, from problem selection to implementation. Noteworthy in this context is the fact that education seems to matter, according to a study by Dologite et al. (1994). The study identified a profile of an individual knowledge based system developer who develops innovative and useful systems. That profile described a person with a computer science degree

who was introverted.

User training for better knowledge management might very well take a different approach in the future, away from learning the syntax of expert system shells, and more towards knowledge explication and formalization. The knowledgeable end user who learns to describe his or her knowledge in a format similar to rules (instead of cases) and who learns to decompose the knowledge into different concepts, and different levels of reasoning, might provide a much better knowledge base to other organization members, albeit not a machine processable one.

REFERENCES

- Barker, V. and O'Connor, D.E. (1989). Expert Systems for Configuration at Digital: XCON and Beyond. *Communications of the ACM*, 32(3), 298-318.
- Blili, S., Raymond, L., & Rivard, S. (1996). Definition and Measurement of End User Computing Sophistication. *Journal of End User Computing*, 8(2), 3-10.
- Coenen, F. & Bench-Capon, T. (1993). *Maintenance of Knowledge-Based Systems*. London: Academic Press.
- Crosby, P. (1984) *Quality without Tears: The Art of Hassle Free Management*. New York: McGraw-Hill.
- Date, C.J. (1990) *An Introduction to Database Systems* (5th edition). Reading: Addison-Wesley.
- Dologite, D, Mockler, R.J., Ragusa, J.M., Lobert, B., & Banavara, N. (1994). Is There a Successful Knowledge-Based System Developer Profile? *Heuristics*, 15(2), 73-83.
- Feigenbaum, E., McCorduck, P., & Nii, H.P. (1988). *The Rise of the Expert Company*. New York: Times Books.
- Francioni, J.M. & Kandel, A. (1988). A Software Engineering Tool for Expert System Design. *IEEE Expert*, 33-41.
- Gaines, B.R. (1987). An Overview of Knowledge-Acquisition and Transfer. *International Journal of Man-Machine Studies*, 26, 453-472.
- Gallagher J.P. (1988). *Knowledge Systems for Business*. Englewood Cliffs, NJ: Prentice Hall.
- Griesser, J. & Tubalkain, T. (1992). Building a Distributed Expert System Capability. in Turban, E. & Liebowitz, J. (eds.) *Managing Expert Systems*. Harrisburg: Idea Group Publishing.
- Hayes-Roth, F., Waterman, D.A., & Lenat, D. (1983). *Building Expert Systems*. Reading: Addison-Wesley.
- Hicks, R.C. (1994-95). Deriving Appropriate Rule Modules for Rule-Based Expert Systems. *Journal of Computer Information Systems*, 20-25.

- Holsapple, C.W. & Raj, V.S. (1994). An Exploratory Study of Two KA Methods. *Expert Systems*, 11(2), 77-87.
- Huntington, D. (1989). Real Time Process Control Expert System Implementations. *AI Review*, Menlo Park: American Association for Artificial Intelligence.
- Irgon, A., Zolnowski, J., Murray, K.J., & Gersho, M. (1990). Expert System Development: A Retrospective View of Five Systems. *IEEE Expert*, 25-40.
- Johnson, P.E. (1983). What Kind of Expert Should a System Be. *Journal of Medicine and Philosophy*, 8, 77-97.
- Lewicky, P., Czyzewska, M., & Hoffman, H. (1987). Unconscious Acquisition of Complex Procedural Knowledge. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13, 523-530.
- Lewicky, P., Hill, T., & Bizot, E. (1988). Acquisition of Procedural Knowledge about a Pattern of Stimuli that Cannot Be Articulated. *Cognitive Psychology*, 20, 24-37.
- Mallach, E.G. (1994). *Understanding Decision Support Systems and Expert Systems*. Boston: McGraw-Hill.
- Mathews, R.C., Buss, R.R., Chinn, R., Stanley, W.B., Blanchard-Fields, F., Cho, R.-J., & Druhan, B. (1989). The Role of Implicit and Explicit Processes in Learning from Examples: A Synergistic Effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15, 1083-1100.
- McCall, J.A., Richards, P.K., & Walters. (1977). G.F. Factors in Software Quality Assurance. *RADC-TR-77-369*, Rome Air Development Center, Rome.
- McNurlin, B. (1987). A Three Year Strategy for Expert Systems. *EDP Analyzer*, 25(3).
- Myers, G.J. (1978). *Composite/Structured Design*. New York: Van Nostrand Reinhold.
- Nonaka, I. & Takeuchi, H. (1995). *The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation*. New York: Oxford University Press.
- Olson, D.L., Schellenberger, R.E., & Mechitov, A.I. (1995). Teaching Knowledge Base Consistency and Completeness. *Journal of Computer Information Systems*, 7-12.
- Panko, R. (1998). What We Know About Spreadsheet Errors. *Journal of End User Computing*, 10, 15-21.
- Parnas, D.L. (1972). On the Criteria to Be Used In Decomposing Systems into Modules. *Communications of the ACM*, 15(12), 1053-1058.
- Perruchet, P., Gallego, J., & Savy, I. (1990). A Critical Reappraisal of the Evidence for Unconscious Abstraction of Deterministic Rules in Complex Experimental Situation. *Cognitive Psychology*, 22, 493-516.

- Reber, A.S., Kassin, S.M., Lewis, S., and Cantor, G.W. (1980). On the Relationship between Implicit and Explicit Modes in the Learning of a Complex Rule Structure. *Journal of Experimental Psychology: Human Learning and Memory*, 6, 492-502.
- Reber, A.S. & Lewis, S. (1977). Toward a Theory of Implicit Learning: The Analysis of the Form and Structure of a Body of Tacit Knowledge. *Cognition*, 5, 333-361.
- Stockdale, A. & Wood, M. (1992). Building a Small Expert System for a Routine Task. *Management Decision*, 30(3), 46-49.
- Tversky A. & Kahneman D. (1974). Judgment under Uncertainty: Heuristics and Biases. *Science*, 185, 1124-1131.
- Ullman, J. (1980). *Principles of Database Systems*. Stanford: Computer Science Press.
- Waterman, D. A. (1986). *A Guide to Expert Systems*. Reading: Addison-Wesley.

Chapter 4

Designing End-User Geographic Information Systems

Lawrence A. West, Jr.
University of Central Florida

Geographic information systems are becoming more popular for end-user and decision support system construction, but they incorporate software and concepts with some inherent problems for users not trained in concepts of geography and cartography. This paper identifies those concepts most needed for end-user GIS use, and suggests remedial efforts to reduce the burden of system operation and improve data integrity. The approaches make extensive use of metadata storage and may be implemented as tools in GIS software provided to end-users.

... designers should start with a clear sense of and respect for the tasks that end-users will be doing, and then design a system that best supports those tasks.

(Bonnie Nardi, 1993, p. 125).

Geographic information systems (GISs) are a specialized class of database management systems that allow users to analyze, relate, and display spatial attributes of data in addition to conventional relational data. While best known for their ability to display data in maps, GISs also provide extensive analytical capabilities, especially for the analysis of data based on geographic location. In many ways, the use of GIS software by end-users parallels the end-user experience

in general, except that there is a decade-long lag between the two. The technology to deliver computerized spatial analysis and mapping to end-users has only been available since the early 1990s, and some of the most important end-user oriented technologies have only been introduced in the past three years.

In one sense, most previous writings on end-user computing (EUC) apply to end-user GIS use. GIS are computer programs that promote individual or group decision making, use data (including data in client/server or data warehouse environments), and have interfaces. Further, GIS technology may or may not be appropriate to the task at hand. Research in these areas will, in general, be applicable to the relevant aspect of GIS usage. This is not to say, though, that previous writings on end-user computing provide a complete prescription for the use of GIS by end-users. In particular, there is a gap in the end-user literature that is unique to the use of GIS.

Effective GIS use for decision making requires expertise in the user's decision-making domain, basic computer skills (file management, editing, etc.), and database skills, all well researched themes in modern end-user computing. GIS use also, however, requires the application of specific knowledge from the fields of geography and cartography, instances of "specific skills" as discussed by Aggarwal (1998). In the past, when GIS users tended to be scientists, engineers, city planners, etc., this specialized knowledge was included in the user's domain knowledge. As GISs become more widely applied to decision making in businesses and governments—where the domain expertise does not include the necessary cartographic knowledge—the ability of end-users to make effective use of these tools is limited.

This paper addresses this issue by providing prescriptive guidance for the support of end-users of GIS technology.¹ The next section provides some additional background on GISs and discusses the relevance of GISs to end-user computing. This section concludes with a list of four specific issues related to the use of GISs by end-users for which specific support is required. The following section presents an overview of the use of metadata as a tool for supporting end-user use of GISs and the final four sections discuss each of the specific problem areas in detail.

GIS AND END-USER COMPUTING

This section has three important roles, each of which is presented in its own subsection. First, an overview of GIS technology is presented. This overview introduces the technology to those unfamiliar with its use, and serves as a necessary foundation for the identification of the specific end-user support issues discussed later in the paper. Examples presented here are referenced throughout the paper. The next subsection expands on the introductory remarks regarding the pertinence

of existing end-user computing research to GIS use. It shows why GISs are a relevant concern for those supporting end-user computing and highlights appropriate literature. The final subsection uses the earlier discussions to derive the four areas of concern for designers of end-user GISs. These four areas then form the basis of the specific discussions that follow.

Introduction to GIS

GISs are specialized database management systems that allow for recording both conventional attribute data in relational tables and information about the spatial location of each record (occurrence) in a table. Figure 1 illustrates the data relationships for one *coverage* or *theme* in a GIS. On the right is the attribute data that can include most types of fields normally considered in a relational database management system (RDBMS) including numeric, text, date, boolean, etc. On the left is a depiction of the location occupied by each object, record, or occurrence. These locations can be polygons (as shown) such as city, county, or state boundaries; linear features such as roads, rivers, or railroads; or point features such as street addresses, building locations, or telephone poles.

The spatial and attribute data are typically kept in separate storage structures but the GIS software can keep track of the 1:1 relationship between each record's attribute data and its positional data so the separation of the data is transparent to the user. Because most modern GISs use conventional table structures for their attribute data, the separation also allows the attribute data to be joined to conventional RDBMS tables that do not have spatial attributes. For example, if a GIS theme has a field for county name and a table in another database also has a field for county name, a *join* can be constructed between the common fields of the two tables. Given appropriate connectivity (e.g., with open database connectivity (ODBC) or client/server technology), the *join* can originate either within the GIS or within the RDBMS.

Figure 1: Spatial and Attribute Data in a GIS Coverage

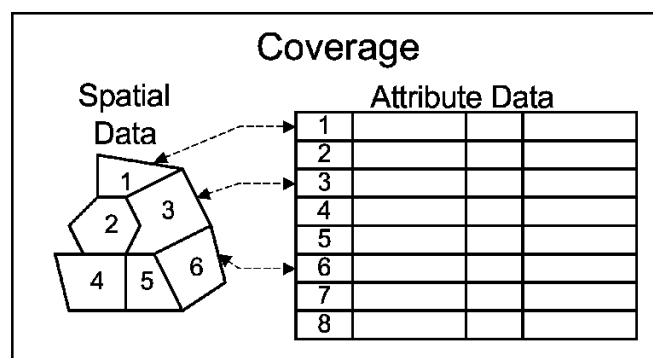
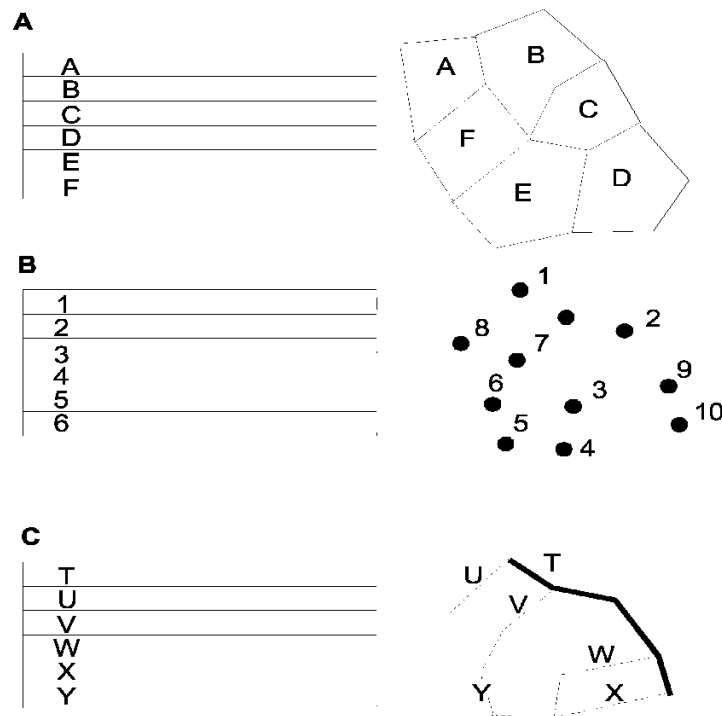
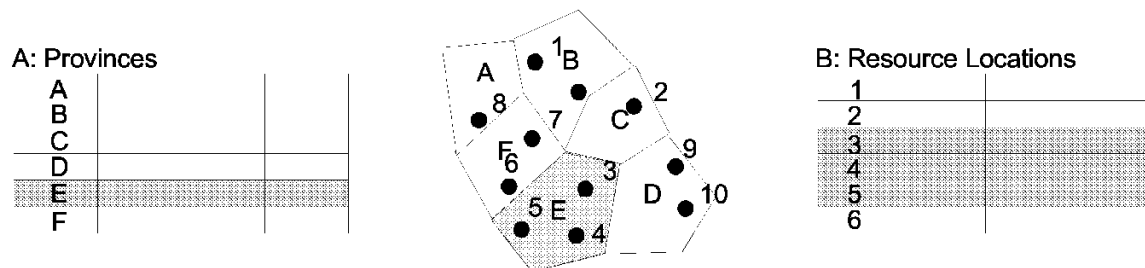


Figure 2: Multiple GIS Coverages*Figure 3: Intersecting GIS Coverages*

The most powerful capabilities of GISs are the ability to display the spatial locations of records on a map and the ability to select records based on their spatial locations. The spatial locations of objects are stored as coordinates using either vector or raster systems² and the GIS 'engine' is 'aware' of the relative locations of the various objects. For example, Figure 2 illustrates three coverages that may exist in a government agency GIS. Coverage A may be regional boundaries, Coverage B may be the locations of exploitable natural resources, and Coverage C may represent actual or potential railroad lines.

Figure 3 then illustrates how the GIS engine is capable of determining that resource locations 3, 4, and 5 are located within the boundaries of Region E. Further, the engine is able to create a join between the attribute (tabular) data for the corresponding tables. It is therefore possible to create a query based on the

spatial proximity of objects. In the present example, the system could sum the annual production for Region E (assuming that annual production was an attribute of the resource coverage) or sum the production for all regions with the results grouped by region. The ability of the GIS engine to create these spatial *joins* gives these systems the same capabilities for summarizing and exploring data that conventional DBMS have except that geography serves as the key between tables (coverages).

End-Users and GIS

Modern PC-based GISs support all of the trends that Aggarwal (1996) identified as being relevant to EUC in the 1990s and beyond (Table 1). GISs have been shown to be particularly well suited to certain classes of decision support (Crossland, 1995; Grupe, 1990) but, while GIS software has been available since the 1980s, only recently has it become widely available to end-users. Early versions required either mainframe or engineering workstation-class computers and did not

Table 1: End-User Trends and GIS Characteristics (Aggarwal, 1996)

Aggarwal's Trends	GIS Characteristics
Organization-wide data	Provide access to organization-wide GIS data through client/server technologies, ODBC, etc. GIS extensions to DBMS such as Oracle, SQL Server, or DB2 support data warehousing and enterprise-wide GIS applications while supporting end-user access through client/server technologies.
UNIX/Windows operating systems	End-user GISs run on both UNIX and Windows platforms with one manufacturer selling a popular product in these and Macintosh platforms.
Multimedia capabilities	Multimedia responses can be triggered by hot spots on a map, events in code or script, or controls in the application window.
Public and private databases	GISs support ODBC connections to a wide variety of data. Typical environment includes in-house produced coverages, coverages (such as street maps) produced by private parties, and coverages (such as census data) produced by government agencies.
Internet access	GIS maps <i>and</i> interactive applications can be delivered via the Internet. Local client applications can use GIS data delivered via the Internet.
Model generators	Support modeling tools such as network problem solvers (shortest route, minimal spanning tree) that use street maps as the network
GUI	All modern PC-based GIS tools use GUI.
Group interaction/ Group decision units networks.	All support exchange of data, stored maps, and analysis through local and wide area
User literacy	GISs require domain and IT literacy but also require geographic and cartographic literacy.

support data sharing. In the early 1990s though, GIS software built to run on PC class computers was introduced, and these products have now evolved to include powerful analysis tools such as those included in high-end PC spreadsheet and database packages. The tools include reporting, graphing, custom query authoring, and, of course, mapping, and represent the collection of task-specific operations that allow users to leverage existing task-related skills (Nardi, 1993). Core capabilities of these systems are also embedded in end-user-oriented products such as Microsoft's Excel spreadsheet program and Seagate's report generator, Crystal Reports.³

Of interest in this paper is the set of capabilities that must be present for end-users to effectively use GISs for decision or task support. Several researchers⁴ have identified domain and computer literacy to be necessary elements of effective end-user computing. Because geographic and cartographic concepts are important for GIS use, these skills must also be present before effective end-user GIS use can be accomplished.

Nardi (1993) highlighted the importance and prevalence of systems of formal notation in everyday life and in domain-specific EUC activities. As previously mentioned, early GIS adopters tended to be scientists, geographers, and others, for whom geographic concepts and the associated systems of notation were part and parcel of their domain expertise. As GIS use spreads to additional circles, however, it finds users in fields such as MIS, marketing, transportation and shipping, scheduling, etc., for whom these important geographic concepts are foreign. Further, Frank and Mark (1991) point out that geographic concepts held by those untrained in geographic principles may not translate well to formalized query languages implemented in many GISs.

Managers with responsibility for EUC in an environment where GISs can be profitably employed must determine if and how the knowledge gap for effective use is to be overcome (Shayo et al., 1999). As Nardi writes, "One piece of the solution to the productivity puzzle then, is to increase the value of computers by increasing the efficiency and effectiveness with which ordinary people use computers to build the applications they need" (1993, p. 123).

One approach to overcoming the skill gap is training. Aggarwal (1998) describes concepts of training in support of EUC that are certainly applicable to end-user GISs. In particular, the specialized geographic and cartographic concepts needed would be addressed as "specific skills" under Aggarwal's definition.

An alternative approach is to embed certain capabilities in the end-user tools themselves. Several writers suggest the careful employment of system enhancements to support user efforts. At one extreme are Rockart and Flannery's (1983) "nonprogramming end-users" for whom all system functionality is embedded in a packaged system. Of more interest is a concept from recent EUC literature, the

provision of specific support for end-users who have a more flexible interaction with the system. Nardi (1993) advocates having a collection of “task specific primitives” available that flatten the user’s learning curve. She points out that the popularity of spreadsheets stems from the relatively few (compared with conventional programming languages) primitives needed to perform important tasks. Gammack advocates providing a “...loose coupling of generic components, from which specific developments can evolve both rapidly and suitably” (1999, p. 16). Morch proposes providing “...end-user developers with easy-to-use yet powerful tools that will allow them to evolve their applications towards new (possibly unforeseen) task domains by incrementally adding task specific building blocks” (1998, p. 25). Finally, Shayo et al., summarize existing research on EUC success and point out that providing software library services and development assistance have been cited as determinants of EUC success. The theme of these works is that common tasks can be preprogrammed and provided to end-users through an easy-to-use interface. End-users may perform tasks through a menu or with wizards rather than through code.

This paper discusses how such support for end-user GISs can be provided through a collection of tools that directly address GIS-specific impediments to effective system use. It is not that training is considered an irrelevant option, it is just beyond the scope of this paper.

Essential Tasks for User Support

Previous experience developing a GIS for end-user support (West, 1999) led to the identification of four specific GIS-related tasks that would be problematic for the average end-user. Each of these tasks was either unique to the use of GIS or was especially problematic because of the GIS. Table 2 lists and briefly explains these tasks, which, along with specific remediation efforts, are discussed in more detail in the sections that follow.

Table 2: GIS-Specific Tasks

Identifying Relevant Coverages	In a system or environment containing a large selection of GIS coverages, it will not always be clear which coverages should be considered or queried for a specific decision-making task.
Layering Coverages	When coverages are displayed on a monitor or printed map it is important that denser coverages lay under less dense coverages so as not to mask the sparser coverage. A related task is managing resolution-dependent display of coverages.
Editing/Adding Data	Data integrity is a key issue in any database system and GISs are database systems. End-user access to the system must protect the data against both conventional data integrity problems but also against GIS-specific problems such as records with overlapping locations if the overlap doesn’t make sense.
Reporting Results	Creating maps is more complex than just printing what displays on the screen. Deciding on the appropriateness of several map components and placing these components can be problematic for inexperienced users.

Metadata Support for End-Users

Metadata has already been shown to be a key factor in providing end-user access to data warehouse collections (Corcoran, 1995; Glassey, 1998) and has also been identified as being useful for access to GIS data (Frank and Mark, 1991; West, 1999). In the discussion of the four problematic tasks (Table 2) in the following sections, it will be seen that each mitigation approach relies to a greater or lesser extent on metadata about the GIS coverages. This section, then, describes the use of metadata to support end-user GIS use. Examples in this section are taken from the author's experience with the FMRIS system (see Endnote 1) and will be referenced in the following sections.

While database documentation is assumed for a well-designed conventional organizational database, the author has found that the majority of GIS data sets are not created by MIS professionals. One of the most striking results of this tendency is the lack of formal documentation standards accompanying GIS data. Further, because many GIS systems use data produced both in-house and by external agencies, it is very likely that documentation, when it exists, will adhere to different standards for naming, etc. The creation of GIS metadata, then, is likely to require specific commitment by those tasked with supporting end-user GIS activities.

Metadata input requires a specialist who is familiar with both the predicted end-user domains, organizational documentation standards, and geographic principles. This person's activities should be supported by a catalog system feeding a metadata repository. The input screen shown in Figure 4 allows input into the table

Figure 4: The Coverage Metadata Input Screen

Coverage Processing

Coverage Input File Name: SHIPWREC

Use this form to enter Coverage data to the Coverage table. The table is used to display options to the user, to record technical and administrative information about the management of coverages and to record other information about the data.

Name: Test Shipwreck Feature Type: Point

Description: Test coverage for illustration. Based on shipwreck coverage

Producer: Unknown

Source: FMRIS

Date Added: 10/01/95 Feature Density: Infrequent Points

Date Updated: Minimum Resolution: 10000 Path Name: shores\ Data Key: Color:

Maximum Resolution: 1500000 Label Field:

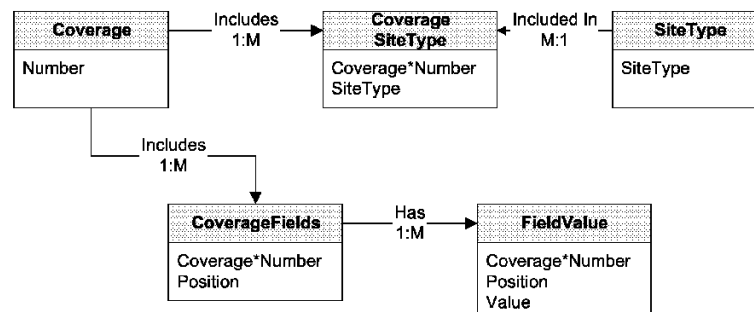
Help Done Accept Fields

Enters the data supplied by you in the various fields. Validates that data type is correct.

Table 3: Structure of the Coverage Metadata Table

Name	Type	Size
Number	Number (Long)	4
Name	Text	20
Description	Text	250
Type	Text	10
Producer	Text	60
Source	Text	60
LastUpdate	Date/Time	8
LastImport	Date/Time	8
Contact	Yes/No	1
MinRes	Number (Long)	4
MaxRes	Number (Long)	4
Level	Number (Integer)	2
FileName	Text	8
PathName	Text	30
pLoad	Yes/No	1
pTransparent	Yes/No	1
pDiscriminator	Text	20
pDiscrimPartial	Yes/No	1
defRed	Number (Integer)	2
defGreen	Number (Integer)	2
defBlue	Number (Integer)	2
LabelField	Text	20

Figure 5: Metadata Portion of the FMRIS LDM



structure documented in Table 3 and will be referenced in later sections of the paper.⁵ The overall database structure for the GIS-specific portions of the metadata repository is shown in Figure 5.

IDENTIFYING RELEVANT COVERAGES

This GIS-specific issue is analogous to the problem of selecting tables and fields for use in a conventional database management system. As with a conventional system, the user's familiarity with the data, its organization, and content, is critical for successful end-user access (Rockart and Flannery, 1983). As Palvia (1991) writes, "...irrespective of what the physical and global logical organization of the data is, the designer should strive to provide the local interface (sub-schema)

of the database to be consistent with the end-user's perception of reality." There are two classes of user actions resulting in the need to load or use one or more specific geographic coverages: accessing a specific coverage directly and accessing a specific coverage based on its contents.

Accessing a Specific Coverage

Many times the user will want to work with a specific coverage. For example, the user may need the coverage of customer service center locations. There are two problems with this task, however. First, many GISs store coverages in directories rather than individual files or named tables within a comprehensive DBMS. GIS coverages typically require a number of individual files to store the spatial and attribute data and these files are stored in a directory with one coverage per directory. Second, naming schemes for GIS coverages tend to be haphazard. Many coverages are not prepared by MIS professionals and many others are produced by external agencies. The naming schemes therefore tend to be inconsistent and cryptic.

The solution to this problem presents the first use of the previously discussed metadata. The **Coverages** metadata table (Table 3) contains four fields, **Name**, **Description**, **FileName**, and **Path**, that can be used to overcome these problems. **Name** stores a 20 character name for the coverage that is significantly more meaningful than the cryptic names that came with the imported coverages. **Description** stores a longer description of the coverage's contents. These two fields allow the display of meaningful coverage names along with expanded coverage descriptions available to the user on demand. In the FMRIS system (see Endnote 1), for example, the names of the available coverages were displayed to the user in a list. The user could highlight a name without selecting it and see the contents of that coverage's **Description** field displayed on the selection screen. When the user selects a particular coverage the **Path** and **FileName** fields are used by the system to find and load the coverage.

Accessing a Coverage by Content

As with many conventional DBMS, the user may want to load coverages based on their contents or applicability without knowing exactly what coverages (tables) would be suitable in a given situation. The **SiteType**, **CoverageSiteType**, and **Coverage** tables (Figure 5) provide critical information to allow users to load coverages based on content. The **SiteType** table lists key concepts of interest to users of the system in terms relevant to the end-user's decision-making domain. It also contains a lengthier description of each concept of interest for use in the user interface just as the **Name** and **Description** attributes (described in the previous section) are used. When the user selects concepts of interest from the **SiteType**

Table 4: Coverage Densities

Polygon Coverages	Line Coverages	Point Coverages
Universal Polygons	Dense Lines	Dense Points
Dense Polygons	Moderately	Moderately Dense Points
Moderately Dense Polygons	Dense Lines	Sparse Points
Sparse Polygons	Sparse Lines	

table, the system uses a conventional SQL query using the **CoverageSiteType** table and the **Path** and **FileName** attributes of the **Coverage** table to determine which coverages to load.

LAYERING COVERAGES

While the ability to create spatial *joins* between geographic coverages provides an important data analysis capability in GIS (West and Mennecke, 1998), some of these systems' strongest contributions come from their ability to present visual relationships between geographic objects (Crossland, 1995; Grupe, 1990). Consider again, though, the coverages illustrated in Figures 2 and 3. If the coverages in Figure 2-A were loaded into a visual display on *top* of those in Figure 2-B then the point locations would be hidden by the polygons. It is important, therefore, for coverages to 'layer' appropriately to avoid masking less dense coverages with denser ones.

The **Feature Density** entry in Figure 4 allows the person mounting the software to indicate the relative density of the features in the coverage. Table 4 lists specific density values for the three types of coverages with **Universal Polygons** being the most dense and **Sparse Points** being the least dense. Values selected during metadata input can be translated into integer values and stored in the **Level** field of the **Coverage** table (Table 3). When coverages are loaded by the system, the **Level** field can be used to load the most dense coverages on the bottom of the 'stack.'

EDITING/ADDING DATA

Data integrity is an important concern in any end-user computing environment (Alavi and Weiss, 1985; Henderson and Treacy, 1986; O'Donnell and March, 1987), and end-user GIS pose some special integrity problems. First, command level use of GIS data through GIS software provides generally unfettered access to the underlying data, including spatial data. Whereas modern database management systems provide many features for enforcing data integrity, these features are often lacking in GIS software. Users can easily change both the attribute data and

the spatial data and, in so doing, violate data integrity standards. Changing shape data can be especially insidious as users may become used to manipulating shapes drawn on the screen and may not realize that they are now changing the locations, vertices, or sizes of shapes from a coverage. These tasks are generally accomplished with deceptively easy-to-use tools that resemble the tools available in many modern graphics programs.

There are four approaches that can be taken to help manage data integrity in an end-user GIS environment: using copies of critical data, using a GIS system that enforces data integrity, restricting user access to data editing features, and constructing specialized integrity checkers.

Using Copies of Critical Data

One of the simplest solutions to the data integrity problem is to provide end-users with copies of critical data. Just as data warehousing separates the warehoused data from the production databases, this approach precludes the user from corrupting the central repository. There are drawbacks, however. Data concurrency problems can develop as different users manipulate copies of the data. Some users may have changes that *should* be posted to the central repository but organizational policies may preclude the updates. Therefore, care should be taken to ensure that data access policies and user permissions are in alignment with user tasks and data use.

Using Advanced GIS Systems

Some companies are now producing GIS extensions to modern relational database management systems (RDBMS) such as Oracle, SQL Server, and DB2. In these systems, the attribute data is stored in the RDBMS and the shape data is stored in special storage structures to achieve the conceptual result illustrated in Figure 1. Of particular interest is that the tools available for enforcing data integrity in the RDBMS can apply to the attribute portion of the GIS data. These rules include enforcing existential and referential integrity, managing permissions, executing triggers, and enforcing domain rules.

There are, however, several drawbacks to these systems. First, they require the investment in a powerful RDBMS, typically running on a mainframe or minicomputer and this investment may be more than some organizations care to make. Second, these systems still do not enforce integrity constraints on the shape data. West and Mennecke (1998) identified several GIS-specific data integrity rules that should be managed in a GIS.

Restricting User Access

It may be possible to restrict end-user access to data editing features of GIS software. If the interface is customizable as it is with ESRI's popular ArcView™

GIS software, then access to both spatial and attribute editing may be disabled for certain classes of users. If the desired or available package does not support restricted access, then the organization must adopt one of the other strategies discussed in this section, a new package must be acquired, or the risks of data integrity violations must be accepted.

Specialized Integrity Checkers

It is possible to construct specialized data integrity checkers that do not rely on stored constraints in the RDBMS. In *n*-tier applications, for example the middleware often contains data management capabilities that may enforce data integrity rules independently of the RDBMS. The FMRIS **CoverageFields** table (Figure 5) contained data integrity rules for key attribute data fields. When users added or edited attribute data, extensive domain integrity checks were performed before the data was posted. If a rule was violated, then the user received an explanation of the problem and the opportunity to correct it or cancel the transaction.

Enforcing data integrity rules for spatial data is more problematic and has not, to the author's knowledge, been implemented yet. West and Mennecke (1998) developed rules for specifying spatial integrity and these rules are not, in principle, difficult to enforce, however, the enforcement would require the implementation of some specialized middleware that makes extensive use of the *SelectByShape* capability of GIS packages.

REPORTING RESULTS

The final category of user support affects preparing map results of a GIS analysis. While GISs are designed to construct maps as an integral part of their operations there are nuances to this process that can escape the untrained user. To provide effective support for end-user operations, the system should provide support for labeling map objects, controlling scale of display, establishing a color scheme, and placing objects on the map.

Labeling Map Objects

GIS software has the ability to label objects visible on the map by using the contents of a field in the coverage's attribute data as a label. If the attribute data is extensive, has cryptically named fields, or if it is hidden from the user, then selecting a label field may be problematic. The **LabelField** of the **Coverage** metadata table in FMRIS overcame this problem by specifying a label field for each coverage as part of the metadata creation process (Figure 4, Table 3).

Controlling Display Scale

Maps are models of the ground drawn to a scale specified as the ratio of a distance on the map to the distance on the ground. As denominators in this ratio get larger, the area of ground represented by the same size map gets larger and the level of detail that can be usefully depicted gets smaller. For example, a map scaled to fit the city limits of a small town on one page may be able to effectively show all roads in the city. If the map scales to show the entire state, however, the roads depicted would run together giving no useful detail and hiding details of other coverages.

Many GIS allow the user to specify the minimum and maximum scale at which a coverage will display. This capability has the effect of automatically turning coverages on or off as the display zooms in or out. Unfortunately, determining appropriate display scales requires an understanding of the significance of an appropriate scale that may be difficult for inexperienced users to grasp. It also burdens the user with an additional step for each coverage as a map is being constructed.

The **minRes** and **maxRes** fields in the **Coverage** metadata table (Table 3) store the minimum and maximum resolutions at which a coverage should be displayed. User support tools can read these values from the metadata when each coverage is loaded. The GIS then automatically turns coverages on or off as the user zooms in or out with the map.

Establishing a Color Scheme

Most GISs establish random color schemes for spatial coverages. By default, all objects in one coverage are the same color but it is relatively easy for users to also specify a field by which to discriminate between objects in the color scheme. For example, a map plotting store locations could have stores with sales in various ranges display in different colors.

Unfortunately, most GISs do not store color schemes with the coverages. When a map (collection of coverages) is created, the user may retain the default color scheme or customize it, and that color scheme will be retained with the map. That is, if Coverage A is loaded into Map B, then the coverage will be reloaded with the most recently saved color scheme when Map B is reloaded. On the other hand, if Coverage A is loaded into a new map the colors established in Map B are not automatically loaded. Instead, random colors will appear and the user must go through the process of customizing the color scheme again.

In some organizations, it is important to use a consistent color scheme for the same coverages, and in land use planning there are widely accepted coloring conventions to denote different property uses. When a consistent color scheme is

desired, it is necessary to recolor the coverage each time it is added to a new project (map). This necessity places a burden on the user and increases the risk that the alteration will not be performed properly.

Metadata and end-user support tools can be used to overcome this burden. The **defRed**, **defGreen**, and **defBlue** fields of the **Coverage** table (Table 3) store integer values that, together, define a color using the RGB color model.⁶ A separate support tool can automatically apply this color to the coverage when it is loaded into any map for the first time. The issue of discriminating between different objects in the same coverage is solved in a similar way using the **FieldValue** table (Figure 5).

Placing Map Objects

The final topic covers the construction and placing of map objects such as a scale, legend, and inset map. It is possible to provide tools to automate this step for users. While these three objects may not be the most important aspects of user support, it turns out that their creation is not intuitive with some software packages. Automating the creation and placement of these elements provides one more level of support to increase the usability of the GIS software while reducing the burden on the user. The automation can often be implemented by scripting the steps that the user would normally follow and creating a menu-driven initiation process. The scripts can use standard settings for defaults and may or may not support user customization.

CONCLUSIONS

GISs are powerful information management systems with particular suitability for decision support and end-user applications. However, their complexity, their use of complex data, and the need to apply obscure cartographic principles makes their immediate use by end-users problematic. On the other hand, many of the GIS-specific problems for end-users can be mitigated through the provision of special purpose tools to perform many of these tasks. These mitigating steps can be implemented as part of a comprehensive application for end-users, or they can be created as tools for use in more free-form end-user systems.

Regardless of the approach selected, metadata will perform a key role in supporting end-users. Each of the specific approaches recommended here required that an expert provide key descriptive information about coverages when they are added to the database. This one-time overhead step then provides essential information used by the user-support components. Fortunately, the idea of a metadata repository is not at odds with conventional views of user support for data access. The geographic elements of an end-user GIS just increases the complexity of the mounting process slightly.

Next, the construction of the assistance tools also requires additional overhead. A key decision, then, is whether to provide end-user support through an investment in software tools or through user training. While recommending the determinants of this decision is beyond the scope of this paper, it is clear that the capital vs. labor investment is familiar to many managers and offers no particular difficulties, at least as long as the software investments are well understood. It is hoped that this paper contributes to this end.

Finally, there is a rich literature on the support, management, and control of end-user computing in modern organizations, much of which was not specifically addressed in this paper. It is important to realize that much of this literature can apply to GISs if the relevant criteria are met. This paper has focused on the GIS-specific aspects of end-user support. It is felt that addressing these issues can significantly increase the use, and therefore the value, of an organization's investment in GIS technology.

ENDNOTES

- ¹ The guidance presented in this paper was first applied by the author in the development of an end-user GIS-based decision support system called Florida's Marine Resource Information System (FMRIS). Certain of the examples presented in this paper are taken from FMRIS. More information on FMRIS can be found in West (1999).
- ² See Arnoff (1989), Chapter 6, or virtually any good GIS reference for an explanation of the difference. The differences are not important for the purposes of this paper but it is important that a consistent format be adopted in an organizational geographic decision support system.
- ³ Information on these two products may be found at the respective company web sites, <http://www.microsoft.com> and <http://www.seagatesoftware.com>
- ⁴ Aggarwal, 1995, 1998; Gammack, 1999; Morch, 1998; Nardi, 1993.
- ⁵ In addition to the screen shown, additional screens exist for entering data about individual field values in the coverages and to establish a permanent color scheme for each coverage. The details of these screens are beyond the scope of this paper.
- ⁶ FMRIS used a separate input screen (not shown here) for entering these values.

REFERENCES

- Aggarwal, A. K. (1996). End-user computing: Revisited. *Journal of End-user Computing* 8(1), 31-32.

- Aggarwal, A. K. (1998). End-user training—Revisited. *Journal of End-user Computing* 10(3), 32-33.
- Alavi, M., & Weiss, I. R. (1985). Managing the risks associated with end-user computing. *Journal of Management Information Systems* 11(3), 5-20.
- Bedoll, R., & Kimball, C. (1990). The importance of meta-data in mass-storage systems. In *Digest of Papers: Tenth IEEE Symposium on Mass Storage Systems* (K. Friedman and B. O'Lear, Eds.). Los Alamitos, CA: IEEE Computer Society Press.
- Corcoran, M. (1995). The warehouse that Jack built. *Systems Management* 23(9), 24-29.
- Crossland, M. D., Wynne, B. E., & Perkins, W. C. (1995). Spatial decision support systems: An overview of technology and a test of efficacy. *Decision Support Systems* 14(3), 219-235.
- Frank, A. U., & Mark, D. M. (1991). Language issues for GIS. In *Geographical Information Systems: Principles and Applications*, (D. J. Maguire, M. F. Goodchild, and D. W. Rhind, eds.). London: Longman.
- Gammack, J. G. (1999). Constructive design environments: Implementing end-user systems development. *Journal of End-user Computing* 11(1), 15-23.
- Glasse, K. (1998). Seducing the end-user. *Communications of the ACM* 41(9), 62-69.
- Gorry, M., & Scott Morton, M. (1971). A framework for management information systems. *Sloan Management Review* 13(1), 55-70.
- Grupe, F. H. (1990). Geographic information systems: An emerging component of decision support. *Journal of Information Systems Management* 7(3), 74-78.
- Henderson, J. C., & Treacy, M. E. (1986). Managing end-user computing for competitive advantage. *Sloan Management Review* 27(2), 3-14.
- Morch, A. I. (1998). Tailoring tools for system development. *Journal of End-user Computing* 10(2), 22-29.
- Nardi, B. (1993). *A Small Matter of Programming: Perspectives on End-user Computing*. Cambridge, MA: The MIT Press.
- O'Donnell, D. J., and March, S. T. (1987). End-user computing environments—Finding a balance between productivity and control. *Information and Management* 13(2), 77-84.
- Palvia, P. (1991). On end-user computing productivity: Results of controlled experiments. *Information & Management* 21(4), 217-224.
- Rockart, J. F., & Flannery, L. S. (1983). The management of end-user computing. *Communications of the ACM* 26(10), 776-84.
- Shayo, C., Guthrie, R., & Igbaria, M. (1999). Exploring the measurement of end-user computing success. *Journal of End-user Computing* 11(1), 5-14.

West, L. A. (1999). Florida's marine resource information system: A geographic decision support system. *Government Information Quarterly* 16(1), 47-62.

West, L. A., & Mennecke, B. (1999). Relational data modeling in geographic information systems. *Journal of Database Management* 10(2), 27-34.

Chapter 5

Hypermedia Document Management: A Metadata and Meta-Information System

Woojong Suh & Heeseok Lee

Korea Advanced Institute of Science and Technology, Korea

Recently, many organizations have attempted to build hypermedia systems to expand their working areas into Internet-based virtual workplaces. Thus, it is important to manage corporate hypermedia documents effectively. Metadata plays a critical role for managing these documents. This paper identifies metadata roles and components to build a metadata schema. Furthermore, a meta-information system, HyDoMiS (Hyperdocument Meta-information System) is proposed by the use of this metadata schema. HyDoMiS performs three functions: metadata management, search, and reporting. The metadata management function is concerned with workflow, documents, and databases. The system is more likely to help implement and maintain hypermedia information systems effectively.

INTRODUCTION

Today, hypermedia documents are growing explosively in many organizations because of the large number of attempts to develop systems employing intranets or extranets for electronic commerce (EC). These systems include hypermedia documents (hyperdocuments) for supporting organizational tasks. Hyperdocuments employed for such tasks are referred to as organizational hyperdocuments (OHDs). They typically play a critical role in

business, in the form of, for example, invoices, checks, or orders. The maintenance of OHD is becoming a burdensome task; managing their needs is as important to economic success as is software maintenance (Brereton et al., 1998). A hypermedia document—a special type of digital document—is based on the inter-linking of nodes, such as multimedia components, etc. (Nielsen, 1993); i.e., it is an application of hypertext technologies employing multimedia components (Fluckiger, 1995). In contrast to other digital documents, a hyperdocument has links to various nodes, “hyperlinks,” which are used as a path for the navigation.

Most of the previous studies on metadata for digital documents have investigated the topic from a technical perspective, such as information discovery. However, corporate digital documents are closely related to business tasks in an organization. In this context, OHDs typically have complex relationships with both information and business processes. The OHDs can impact the speed of communications and the productivity of business processes. Accordingly, OHDs should be designed to support collaboration among workers in business processes. This aspect needs to be considered in defining metadata of the OHDs for their effective management. Furthermore, such documents should also be considered from a technical aspect. The system resources used by OHDs are a considerable part of the organizational assets.

The two objectives of this paper are (i) to propose metadata classification and metadata schema for OHDs, and (ii) to implement a meta-information system on the basis of the schema. The system was designed to support the maintenance of OHDs. Our research is rooted in previous studies on the various types of multimedia documents so as to capture the more generic perspective of metadata.

METADATA AND META-INFORMATION SYSTEM

Metadata is generally known as data about data (or information about information). Metadata for digital documents has been explored from various research perspectives: mixed media (Chen et al., 1994), multimedia representations (Kashyap & Seth, 1996), document objects (Sutton, 1996), and networked information resources (Dempsey & Weibel, 1996). Much past research has concentrated on the use of metadata to support access to media- and application-specific documents. This metadata describes various system properties, such as video (Jain & Hampapur, 1994), images (Anderson & Stonebraker, 1994; Kiyoki et al., 1994), or speech and text document (Glavitsch et al., 1994). In contrast to these, it has been suggested that media-integrated metadata should be developed for the management of documents

with heterogeneous properties. There have been attempts to do this (Mena et al., 1996; Shklar et al., 1995).

These studies have described metadata roles in various technical aspects from the perspective of document types or system environments. Efficiency in document access control or interoperability of heterogeneous documents has been discussed as the prime problems of these systems. A set of hyperdocument metadata, the Dublin Core (Dublin Metadata Core Element Set) (Dempsey & Weibel, 1996; Weibel et al., 1995; Weibel & Iannella, 1997), has also focused on the information discovery; these are some of the difficulties in managing OHDs (Murphy, 1998).

Metadata of OHDs should be considered beyond the technical aspects by including an organizational aspect toward organizational memory (OM) because they are a major source of organizational memory (Meier & Sprague, 1996; Murphy, 1998; Sprague, 1995). The concept of OM has many facets, but most authors agree that OM must support decisions by using OM techniques for managing an organization's information or knowledge of the past (Shum, 1997; Stein & Zwass, 1995; Wijnhoven, 1998). In this context, a meta-information system for OHDs can evolve toward managing OM by extending their metadata scope to capture their history in terms of business functions, communication mechanisms, or technical artifacts, beyond focusing on contents discovery. These memories may provide knowledge to support various decisions for controlling communication mechanisms in a business process, linking to the previous responsible workers, or maintaining the hypermedia applications. Metadata roles can be summarized in three levels (operation, system, and organization) as shown in Table 1.

Table 1: Metadata Roles

Level	Metadata Roles
Operation	<ul style="list-style-type: none"> • Easy and fast access • Increased accuracy
System	<ul style="list-style-type: none"> • Interoperability under heterogeneous environment • Document maintenance • Document distribution
Organization	<ul style="list-style-type: none"> • Increased reusability of information and knowledge resources • Increased capacity of business management • Increased organizational memory

A meta-information system can be characterized by information resources (to be controlled) or supported services; they service three types of domains: application-oriented, hybrid, or management-oriented.

Application-oriented meta-information systems use metadata to support the application functions. Therefore, metadata schemas are primarily determined on the basis of system requirements. One example is a type of Web search engine. Its main task is to search for information. The main users of this system domain may be application end-users.

In contrast, management-oriented meta-information systems play a major role in supporting the reuse and maintenance of managerial resources. In a document-oriented environment, these systems should serve managerial capabilities for the system- and business-related information or knowledge, through the management of the metadata on organizational documents; major users may be system analysts, information managers, or system administrators.

Hybrid domain systems pay attention to metadata for the managerial purposes, as well as specific application functions. Accordingly, the major users may not only include application managers but also end-users or application specialists. Examples of this domain include EDMS (Electronic Document Management System), which requires metadata as an essential component for document handling (Sutton, 1996).

METADATA CLASSIFICATION AND ELEMENTS FOR HYPERDOCUMENTS

Metadata Classification

Metadata classification can be perceived as a fundamental framework for providing metadata elements. The roles of the elements are determined by the classification coverage. Bohm and Rakow (1994) proposed a metadata classification for multimedia documents. It focuses on the representation of media type, content, relationships among document components, history, and location. Another metadata classification is specified by the dependency on the content (Kashyap & Sheth, 1996; Mena et al., 1996; Shklar et al., 1995). On the basis of these two kinds of classification, metadata for managing medical documentation using hypermedia (Consorti et al., 1996) and quality of service in distributed multimedia systems (Kerherv et al., 1996) have also been developed.

This study proposes the following metadata classification for organizational hyperdocuments:

- *Content-Dependent Metadata*: This metadata is used to enable understanding of the content of documents. The metadata includes information that

depends on (i) the content directly, and (ii) semantic meanings based on the content of the document indirectly.

- *Workflow-Dependent Metadata*: This metadata provides information about workflow related to an organizational hyperdocument.
- *Format-Dependent Metadata*: This metadata describes information on formats related to organizational hyperdocuments, as well as hypermedia components, such as nodes and interface sources.
- *System-Dependent Metadata*: This metadata provides information concerned with storage- and software-related information on system resources, such as hyperdocuments, interface sources, and databases.
- *Log-Dependent Metadata*: This metadata describes information on the history and the status of organizational hyperdocuments.

Workflow-dependent metadata is concerned with process-related factors such as workers, tasks, or business rules. Generally, corporate documents are produced in undertaking an organizational process (Uijlenbroek & Sol, 1997), furthermore, most businesses are based on, or driven by, document flow (Sprague, 1995). Thus documents and business processes may be considered simultaneously in the analysis of a corporate information system (Frank, 1997). Accordingly, workflow-dependent metadata is required for the effective management of OHDs in an information system.

Format-dependent metadata is concerned primarily with the artifacts related to a hyperdocument, such as nodes, anchors, interface sources, or database attributes. The meta-information of formats can provide an understanding of the hypermedia features in terms of structures and operational mechanisms, so that it can be useful in the technical maintenance of hyperdocuments. The system-dependent metadata can also play a critical role in technical maintenance by providing information on hardware and location, and software technologies applied to the hyperdocuments. This meta-information is essential for sharing and reusing system resources. Finally, log-dependent metadata may contribute to organizational memories. Thus, the metadata in this category should be specified in order to capture the history of OHDs.

Metadata Elements

Metadata elements for digital documents have typically been determined differently according to the characteristics of documents and purposes of their systems. Most of the OHDs in business applications typically perform complex functions that are often connected with a corporate database for business tasks in a workflow. This paper focuses on OHD maintenance in consideration of processes and system artifacts. From this perspective,

Table 2: Metadata Elements of Organizational Hyperdocuments

Classifications	Elements
Content-Dependent	[Document] Title, Description, Document Domain Name, Conceptual Attribute Name [Anchor] Name [Data Node] Title [Interface-Source] Name
Workflow-Dependent	Task Domain Name, Task, Agent Domain Name, Agent Object Name, Business Rule
Format-Dependent	[Document] Type [Anchor] Type [Node] Type, [Data Node] Property [Interface-Source] Property [DB] Physical Attribute Type
System-Dependent	[Document] File Name, H/W Name, Location Path, S/W Technology [Data Node] File Name, H/W Name, Location Path [Interface-Source] File Name, Storage, Location Path [Database] Name, H/W Name, Location Path, Table Name, Table Type, Physical Attribute Name, DBMS Name
Log-dependent	Document Number, Version Number, Loading Date, Withdrawal Date, Update Date, Update Description, Director, Operator

detailed metadata elements may be specified under the classification suggested in this paper, as shown in Table 2.

Content-dependent classification consists of elements that enable users to understand the content of the hyperdocuments. The document domain may be in terms of content and roles. The conceptual attributes, as data items represented on a hyperdocument, are connected to a corporate database. Interface sources are primarily multimedia components, such as image or animation that are represented on interfaces.

A node, an essential factor of hypermedia, has been defined as the fundamental unit of hypertext (Nielsen, 1993), fragments of information (Fluckiger, 1995), or basic information containers (Schwabe & Rossi, 1994). This paper defines a node as any navigational object with hyperlinks. An object may be a type of media, such as image, sound, animation, or a hyperdocument itself. Nodes may be categorized into two types from the perspective of their properties: document nodes and data nodes. Nodes are

Table 3: Types of Nodes

Perspectives	Types	Descriptions
Properties	Document Node	A unit of an HTML document, which may be a whole interface or a part of it.
	Data Node	A unit of multimedia data which may be accessed from a document node.
Link Direction	Source Node	Nodes which can access to a current node.
	Destination Node	Nodes to which a current node can access.

also of two types from the perspective of link directions: source and destination. The fundamental definitions for nodes are summarized in Table 3.

An interface may consist of one or more hyperdocuments. Accordingly, a document node, a hyperdocument, can be either only a part of an interface or an interface itself. From these definitions, the element of node type in format-dependent metadata may take a document node or data node as its value.

The information of a hyperdocument in terms of a process can be obtained effectively by the use of a document-based workflow concept. The workflow concept typically includes common essential factors in terms of a unit of a work, a tool of a work, and a person for a work. In the document-based workflow approach, an OHD is regarded as a tool of a work. A task, as a work unit consisting of a workflow, may be described as operations or descriptions of human actions with a hyperdocument. An agent refers to a person who performs the task, and is expressed by hierarchical status in an organization. An agent domain can be defined as a group of agent objects having common tasks or organizational objectives. The agent domain can be typically conceived as a department of an organization. The task domain is a set of tasks corresponding to an agent domain. This metadata can be captured effectively by the use of a document-based workflow model proposed in WHDM (Workflow-Based Hypermedia Development Methodology) (Lee & Suh, 1999).

The format-dependent metadata is concerned with type or properties of hyperdocuments, anchors, nodes, interface sources, and databases. The types of anchors can be static or dynamic depending their value. The definitions of these types are as follows:

- *Static anchor*: One fixed in a hyperdocument.

- *Dynamic anchor*: One generated by data stored in a database; i.e., it refers to data transformed into and represented as an anchor when the data is accessed by a hyperdocument according to any event that occurs as a function or another anchor.

The types of OHDs can be categorized into three: control, processing, and referential, according to their roles in a hypermedia application. These types are defined as follows.

- *Control Type*: Hyperdocuments that typically guide users to other hyperdocuments of processing or referential types. Homepages or index pages are examples of this type.
- *Processing Type*: Hyperdocuments that typically contain data attributes connected with a database in the style of a form.
- *Referential Type*: Hyperdocuments that provide supplementary information about work instructions, business rules, news, or products.

Properties of interface sources are multimedia properties such as images or animation. The properties of data nodes are the same as those of interface sources. The physical attribute type of a database implies the data properties of the attribute.

System-dependent metadata focuses on storage-related information. The storage-related information can be found in various applications, but they are not integrated, so it is difficult to create a synergetic effect. However, if metadata of all the components of hypermedia systems, such as hyperdocuments, data nodes, interface sources, and databases, are integrated into a repository, it is possible to manage a hypermedia system effectively. Software technology is a major factor in determining the capacity and characteristics of a system. Recently, for example, a variety of emerging software technologies, such as ASP (Active Server Page), Java scripts, Visual Basic scripts, or Perl, have had a considerable impact on the improvement of hypermedia systems. Accordingly, the information on software technologies applied to a hyperdocument may contribute to the maintenance of a hypermedia application.

Log-dependent metadata is used for tracing the history of hyperdocuments for the maintenance of their system. Although there may be log information captured automatically by an operating system or an application program, it is typically separated, so it is difficult to obtain a synergetic effect in maintenance. Furthermore, it is insufficient to maintain a hypermedia system effectively. Therefore it is necessary to capture the information about changes of hyperdocuments synthetically. Some hyperdocuments may be operated temporally, depending on their purposes. Accordingly, version- or time-related information should be managed. The loading date is a date pushing a

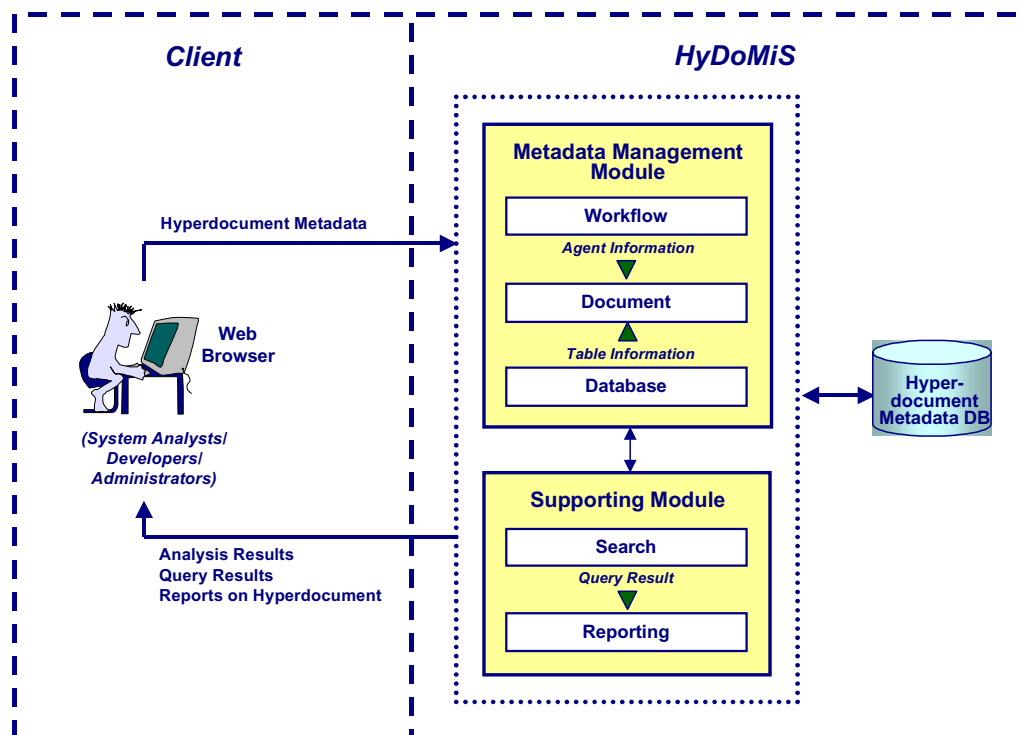
hyperdocument into its system. The withdrawal date is a date removing a hyperdocument from the system for the expiration of its periodic role or for updating. Information on responsible operators and directors for a hyperdocument may be required for responsible management, or questions by new staff members.

HYDOMIS ARCHITECTURE

This section introduces the prototype of a meta-information system for organizational hyperdocuments (OHD), called Hyperdocument Meta-information System (HyDoMiS), included in a management-oriented meta-information domain. HyDoMiS was constructed to manage OHDs effectively through their metadata. This system may affect economic success in maintaining organizational hypermedia applications based on intranet or extranet.

HyDoMiS consists of two main modules: metadata management and a supporting module. These modules have their sub-modules, as shown in Figure 1. The metadata management module is responsible for metadata handling such as creating, editing, or deleting. The supporting module serves two types of functions—searching an OHD and reporting its meta-information. These functions are based on a hyperdocument metadata database.

Figure 1: HyDoMiS Architecture



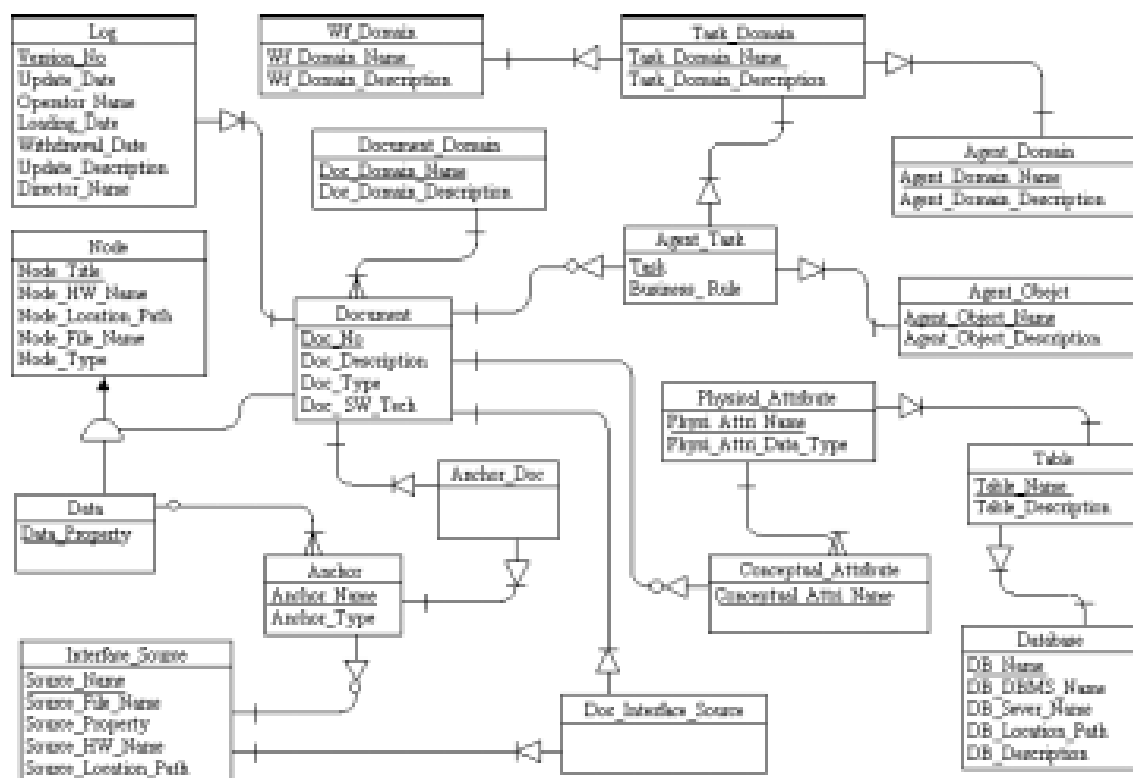
Workflow information enables us to understand the hyperdocuments' roles in a business process. This information is concerned with workflow domains, agents, tasks, and business rules of an OHD. The document metadata management module is composed of five sub-modules, a hypermedia system, data attributes, navigation, interface sources, and log information. System-related information focuses on the hardware and the software technologies. Attribute information is concerned with data provided by a corporate database. This sub-module, therefore, can provide complete information, so long as database-related information is provided from the database information sub-module. The navigation information sub-module provides meta-information in terms of two kinds of nodes (destination node and source node). That is, for a certain hyperdocument, we can get not only information of source nodes which can go to the hyperdocument, but also information about destination nodes which are differentiated into two types, document node and data node. The interface source information is useful to increase the reusability of multimedia components represented on hyperdocuments. The database information module deals with information about databases connected to hyperdocuments. The information produced in this module can be used for the control of the connection between hyperdocuments and a database in the document metadata management module.

The search module provides the numbers and titles of searched hyperdocuments as a result. The search module employs two approaches: drill-down search and keyword search. The drill-down search uses a mechanism to narrow the domain to find a hyperdocument. This search can be performed by the use of various domains in terms of workflow, database, navigation, interface source, and log information. The keyword search uses a typed keyword complying with the selected items, such as workflow name, document title, abstract, or anchor name. The result produced in a search module can be transferred to a reporting module automatically, in order to generate a report on a document search. The report can provide meta-information whose nature will depend on which reporting domains are selected.

The metadata schema is produced based on metadata elements proposed in Table 2. The schema was designed as an E-R diagram for implementing a metadata database for HyDoMiS as shown in Figure 2. The schema was centered on document entity.

This schema represents complex relationships among the components employed in hyperdocument operations, and contains managerial factors for the control of task or system. Such schema content can be captured effectively through the processes of the hypermedia development methodology, WHDM

Figure 2: Metadata DB Schema of HyDoMiS



(Workflow-Based Hypermedia Development Methodology) (Lee & Suh, 1999) rather than other methodologies such as VHDM (View-Based Hypermedia Design Methodology) (Lee et al., 1999a) or SOHDM (Scenario-Based Object-Oriented Hypermedia Design Methodology) (Lee et al., 1999b). WHDM employs a document-based workflow model to capture the requirements for OHDs to be implemented.

A CASE AND A SYSTEM IMPLEMENTATION

HyDoMiS was constructed as a Web server based on Internet Information Server (IIS) 4.0 for multi-users such as developers or system administrators. These users can access the HyDoMiS through Web browsers which belong to the client. The Visual Basic script based on ASP technology was used primarily for implementing functions for dynamic navigation, metadata controls (creating, editing, and deleting), search and reporting. The metadata DB was developed with Microsoft SQL Server 6.5.

In this section, each module of HyDoMiS is illustrated by using a real-life case for a bank in South Korea. The case is concerned with OHDs in a workflow for individual loans that require insurance policies which are issued

by a professional organization for credit insurance. This workflow requires rigorous approval procedures through several steps, because it is important to investigate an applicant's ability to repay, as well as his credit.

Metadata Management

This module takes responsibility not only for storing and retrieving metadata for hyperdocuments but also for providing useful meta-information based on the queries using the metadata. This module includes three sub-modules: workflow, OHDs, and databases.

Workflow Metadata Management

Workflow metadata management module deals with meta-information on constituent factors of workflow, such as workers, tasks, and business rules related to an OHD. Moreover, this information can be reported on the basis of the relationships generated automatically through the procedure of metadata creation. The workflow meta-information is managed at two levels: domain and specification. The domain level manages information on workflows, agents, tasks, and hyperdocuments, which can be guided to their modules from screen A of Figure 3. Among these modules, workflow domain information in screen B can be created in screen C, and can be edited or deleted in screen D. Screen B shows the private loan workflow that is the case already explained. Screen C is linked to the icon of a pencil in screen B, and screen D is accessed from dynamic anchors generated as workflow domain names in screen B. Most of the other sub-modules in metadata management were implemented in this manner for reporting and controlling the meta-information.

The task domain management module generates the relationships between task domains and the other domains concerned with a workflow and agents which are already produced, as reported in screen A of Figure 4. This report makes clear which workflow domain is processed by which agent domains related to which task domains. Screen A lets us see the workflow—individual loan—and task domains; it is controlled by the telemarketing department and loan center. This information may be used to control the roles of OHDs in a business application. In this screen, task domain names are also, like dynamic anchors, linked to their editing screen. Furthermore, the data listed in each domain column can be sorted out by clicking their titles.

The module of workflow specification manages detailed information about OHDs, in terms of agent objects, and tasks, on the basis of the domain information. In this module, the task specification management module provides full specification related to a workflow, as shown in screen B of

Figure 3: Screens for Workflow Domain Metadata Management

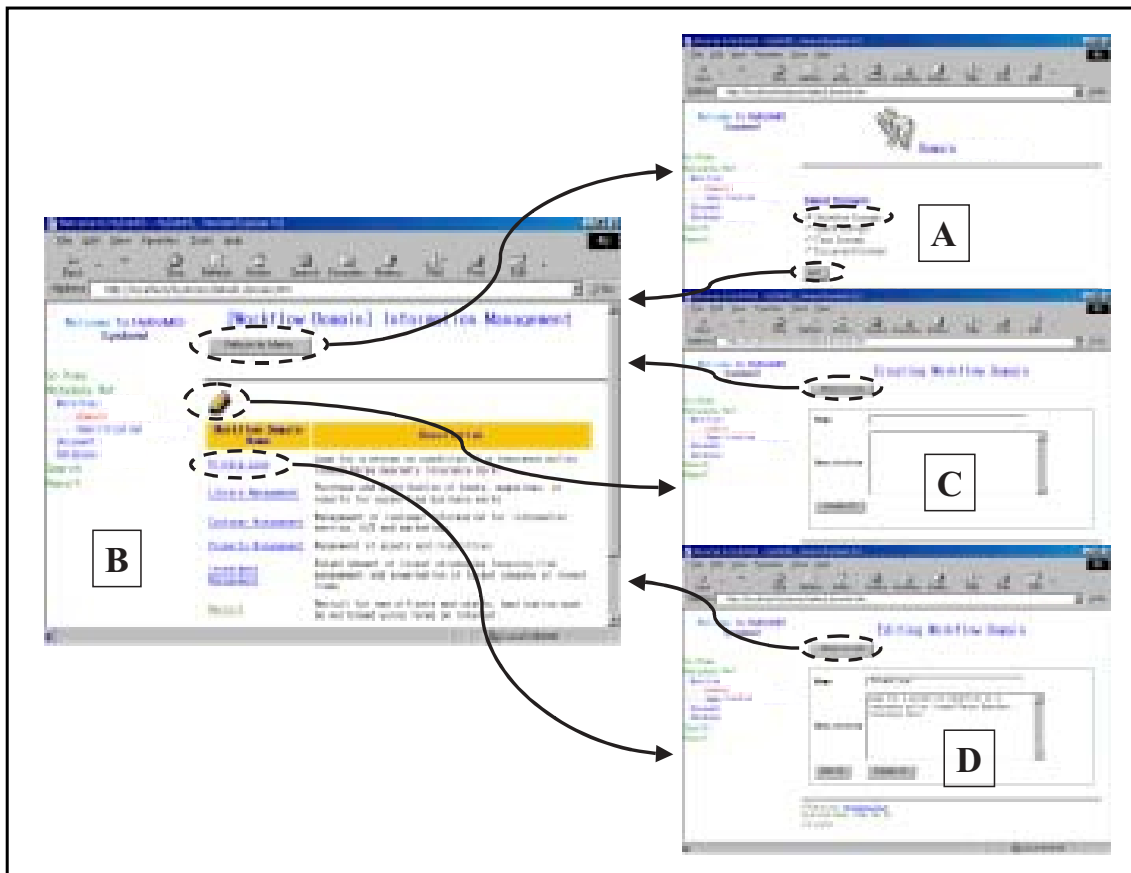


Figure 4: Screens for Task Domain and Task Specification Metadata Management

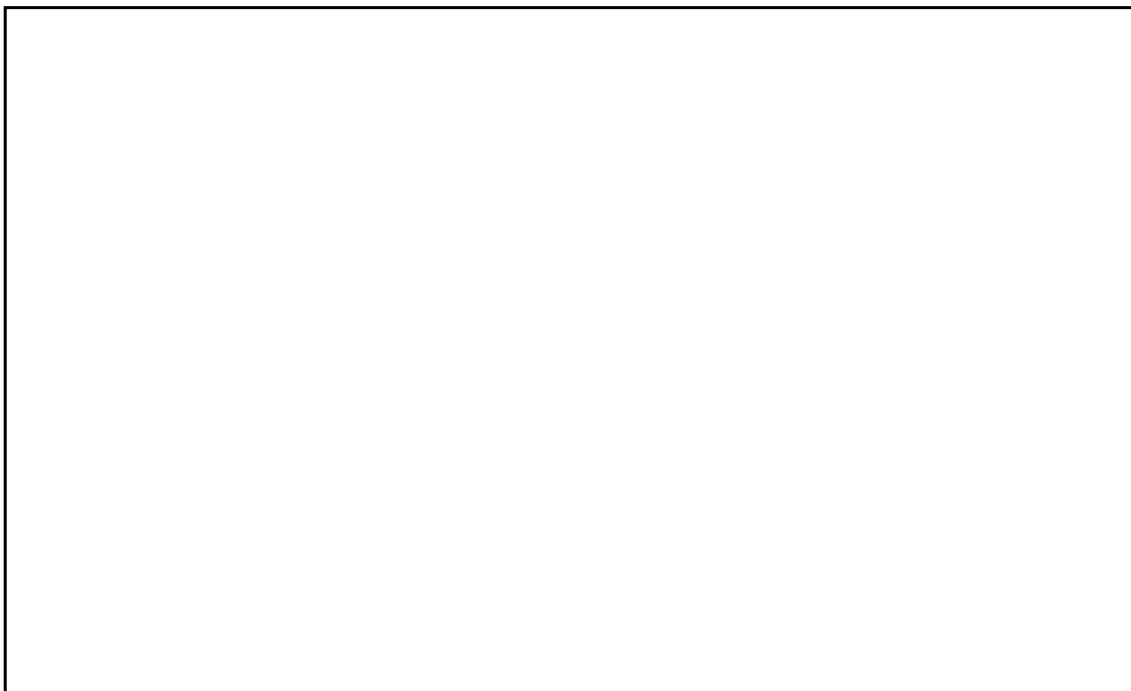


Figure 4. From the report of screen B, for example, in individual loan workflow, LP-1112, "Approval Confirmation," is used by a head (manager) for the task of approving a loan, according to the business rule checking the application information and confirming the applicant's ability to repay within the criteria. Such business rules related to an OHD can be applied differently, depending on agent object's tasks. The information concerning the history of the business rules applied to the agent objects may become a valuable part of organizational memory, and may contribute to improvement of the business productivity. The LP-1112 document is used by credit staffs for the issue of a form submitted to the insurance company in order to apply for an insurance policy required to guarantee the loan application. To provide this task-specific information, the other metadata on agent objects and documents should be created in advance.

Document Metadata Management

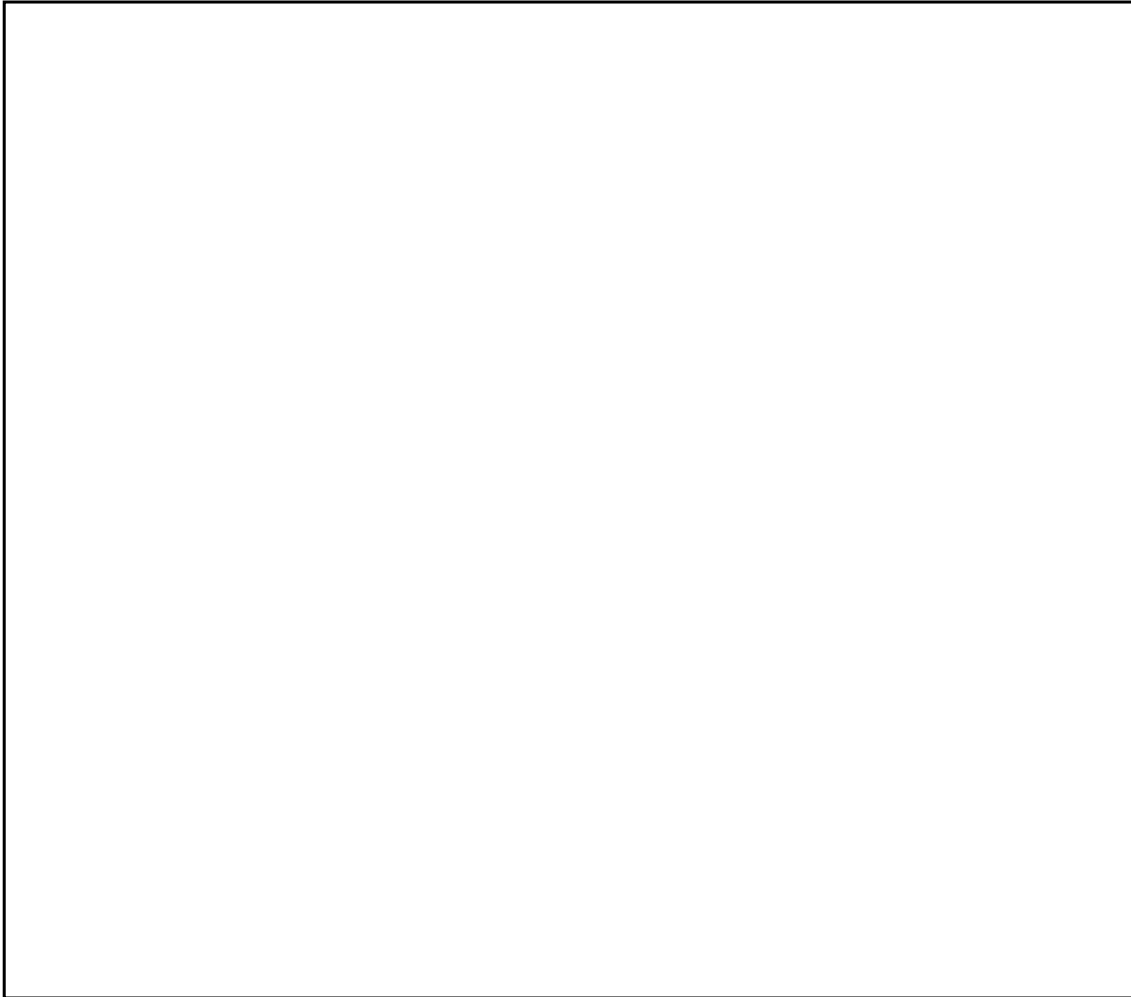
Document metadata management concentrates on the information of hyperdocuments themselves in terms of their content, system-related resources and managerial history. Its module is made up of five sub-modules concerned with storage and software, attributes, navigation, interface sources, and logs. The information provided by these sub-modules is helpful in understanding content, managing system resources, and controlling the navigation relationships of hyperdocuments. Therefore, the maintenance of a hypermedia system can be supported effectively by the use of such information.

Among the sub-modules, a system specification module manages information concerned with storage and software technologies. The information on document number, title, document domain, document type, file name, hardware name, and location path is managed in the module. This information is needed essentially to provide access to a hyperdocument, and to use software technologies for the maintenance of hyperdocuments.

Second, the document attribute management module is implemented for the purpose of controlling the connection of a hyperdocument with a database. Accordingly, this module provides the information needed to manage the relationships between conceptual attributes and database information, including tables and physical attributes. The database-related information is supported by the database metadata management module (See next section).

Third, the navigation management sub-module is responsible for managing nodes and hyperlinks. The hyperlinks, as the most essential component of hypermedia along with nodes, provide navigation paths between nodes (Fluckiger, 1995; Nielsen, 1993). Accordingly, for the effective management

Figure 5: Screens for Navigation Information Management



of hyperdocuments, it is essential to manage information on hyperlinks and their node specifications. This sub-module provides detailed information on nodes from two perspectives: properties and link direction. The information about source node may be useful if a hyperdocument should be changed or removed. Screen A of Figure 5 lets us see information on anchor names and document file locations of hyperdocuments that can access LC-11 – the homepage of the loan center. Accordingly, if the content of LC-11 is changed or the document should be removed, we can edit the anchors of the source node without missing important information. That is, with the help of source node information, system managers can perform maintenance activities effectively through the efficient control of relationships among hyperdocuments.

Screen B of Figure 5 shows the information on document nodes where LC-111 can go, while screen C lets us know the information on data nodes that LC-11 can call for. This module provides location information of the nodes, which may help users control the navigation relationships among nodes.

Fourth, the module managing interface sources is included in the document metadata management module. Interface sources are typically multimedia data represented on a hyperdocument for its rich semantics. They may have properties of some multimedia types, such as image, or animation. It is important to manage interface sources for increasing the representation effectiveness of hypermedia information. This module provides information about source file names and their locations, as well as their properties.

Fifth, the log management module supports management of a document history in terms of time, changed specification, or a person in charge of the document. The information about these considerations is managed through version control. It is generally conceived that hyperdocuments have an advantage of flexibility in accommodating rapidly changing business requirements. Furthermore, the fast development of hypermedia technology has a great impact on motivation for the change of hypermedia applications. Accordingly, more complete log management is required for more effective maintenance of a hypermedia system.

Database Metadata Management

The database metadata management module manages information on a database that supports data transaction of an organizational hyperdocument. The information of a database and its location created in this module is used for providing meta-information of relationships between OHDs and a database related to them. This module consists of two sub-modules: database information and table information.

Search and Reporting

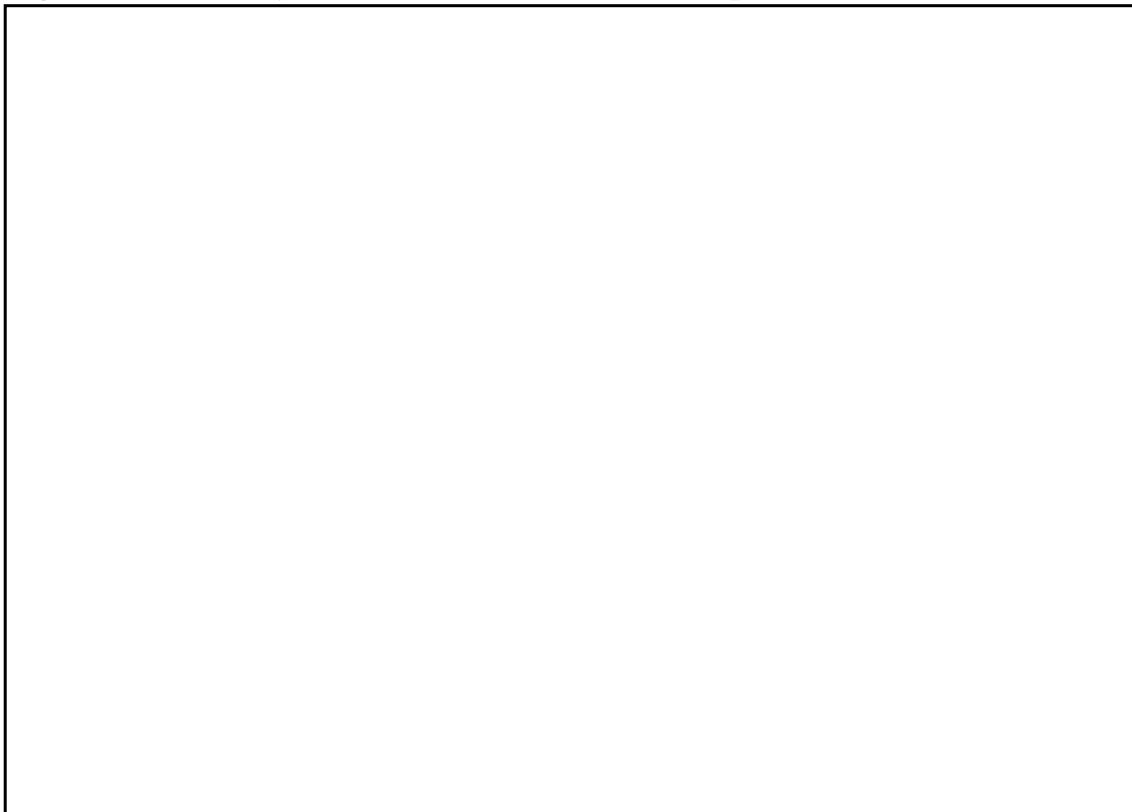
In a hypermedia system, if some components in terms of interface sources, such as logo, database schema, or processes, should be changed, it is not easy to find all the hyperdocuments related to the components in a short time. Accordingly, the search function can play a critical role in maintaining a hyperdocument.

These modules were developed for the purpose of direct maintenance of hyperdocuments by using the metadata stored in a metadata database. The search function was implemented in two ways: drill-down search and keyword search. The drill-down search derives nested attributes from the selected attributes, and the query is based on the attributes chosen from the latest nested ones. The keyword search gets an input word, and finds hyperdocuments related to the word. As a search result, hyperdocument numbers and titles are listed, then if a user selects a hyperdocument, meta-information can be reported on the document depending on the reporting domain chosen.

Drill-down search provides a user with a mechanism to explore hyperdocument lists that belong to search domains by narrowing them down. This search method can provide nested attributes resulting from the query about attributes already selected. As an example, if a workflow domain is selected, then agent domains, task domains and document domains (which are nested in the workflow domain) are queried and listed in combo boxes, as shown in the screen A of Figure 6. The other drill-down domains, which cover all the meta-information domains of an OHD, can be seen in screen A if the screen is scrolled up.

Agent objects or tasks may be queried in the same way. The selected items are listed in the above drill-down search interface. Search results can be represented as the lists in screen B of Figure 6. If a document is selected from the result lists in B of the above figure, then the number of the document is passed to the report module and appears automatically in the input box in the report on screen C. The reporting module provides the options for the searched domain. Screen D shows meta-information about LP-1113, depending on the searched domain option, "All Information." Besides the basic information and system specification in screen D, the other information in terms of workflow, data attributes, navigation, interface sources, and log can be seen if the screen is scrolled up.

Figure 6: Screens for Drill-down Search and Reports



A keyword search may be more useful in search of a hyperdocument by use of a keyword which users already know, even if the keyword is incomplete. If the keyword, for example, is “logo”, the information on interface sources which include “logo” in their names can be listed as two kinds, bank-logo and center-logo. Interface sources may be used repeatedly in many documents. Accordingly, if an interface source is changed or should be removed, the query to find documents including the source may be a useful tool for system maintenance. The keyword search is performed depending on the search options such as workflow name, table name, document name, anchor name, or operator name. The results of a keyword search can be used in generating a report on searched documents in the same manner as the results in the drill-down search.

Implications

Metadata has been employed as a critical means for managing system resources. The need for managing compound multimedia documents using metadata has been pointed out. Hyperdocuments, as a type of compound multimedia document, may have common metadata with other types of multimedia documents. However, for more effective management of hyperdocuments, it is necessary to investigate metadata to capture their own generic features such as the links. Hyperdocument metadata proposed by past research has a limitation in managing OHDs which support organizational tasks. Accordingly, task-related metadata needs to be incorporated into their metadata schema. The HyDoMiS proposed in this paper was developed from this perspective in order to provide meta-information for maintaining OHDs technically and supporting organizational process management.

Practical benefits of the HyDoMiS can be summarized as follows: First, it can efficiently support the changes of OHDs corresponding to the organizational tasks which need to be changed. HyDoMiS can provide information to analyze business tasks and OHDs simultaneously. Second, complex hyperlinks of OHDs can be controlled effectively. Most changes of hypermedia information systems (HISs) are more likely to require handling hyperlinks. Accordingly, the ability to control the hyperlinks is important to maintain HIS. Third, it is possible to effectively control the coordination between hyperdocuments and a database. Meta-information on cooperative relationships between a database and hyperdocuments enables us to take efficient action for the requirements for changing business data. Fourth, system resources related to OHDs can be reused and managed effectively, which can reduce many programmers' efforts for maintenance. For example, if it is necessary to change an interface component, all the hyperdocuments that

contain the component can be found by search and reporting functions. Fifth, organizational memory can be improved from the technical and business perspectives. The history of business requirements and the responses of HIS to these requirements can be accumulated. This history can help new members understand a system and contact past responsible staff members for maintenance.

In spite of this usefulness, HyDoMiS still has a weakness related to automatic problems. For higher efficiency of HyDoMiS, it is necessary to improve the automatic capability for obtaining metadata in two ways: one is to make an automatic coordination possible with a CASE tool which supports development of database applications, and the other is to make possible automatic changes of metadata resulted from the change of OHDs.

CONCLUSIONS

Recently, many organizations have expanded their business by the use of Internet technologies. Organizational hyperdocuments are critical resources for such organizations. Managing the documents may impact the success of business.

In this paper, we propose a meta-information system, HyDoMiS (Hyperdocument Meta-information System), on the basis of a metadata database for the effective management of hypermedia resources. In order to generate a metadata schema for HyDoMiS, a metadata classification for hyperdocuments is studied, and metadata elements are thus specified. The metadata schema is developed from an organizational perspective in terms of processes and technical artifacts. HyDoMiS is constructed as a Web server for a variety of users, such as system analysts, information managers, or system administrators. HyDoMiS is expected to become a repository system for organizational memory, in the long term.

Another contribution of this paper is that it redefines the complex relationships among the components employed in hyperdocument operations, and it captures the definitions in a metadata schema for the HyDoMiS metadata database.

On the basis of the current research, we are in the process of incorporating SGML (Standard Generalized Markup Language) documents into HyDoMiS functions. Another research challenge is to apply HyDoMiS to a futuristic knowledge repository.

REFERENCES

- Anderson, J.T. & Stonebraker, M. (1994). Sequoia 2000 metadata schema for satellite images. *ACM SIGMOD Record*, 23(4), 42-48.
- Bohm, K. & Rakow, T.C. (1994). Metadata for multimedia documents, *ACM SIGMOD Record*, 23(4), 21-26.
- Brereton, P., Budgen, D. & Hamilton, G. (1998). Hypertext: The next maintenance mountain. *IEEE Computer*, December, 49-55.
- Chen, F., Hearst, M., Kupiec, J., Pederson, J. & Wilcox, L. (1994). Metadata for mixed-media access. *ACM SIGMOD Record*, 23(4), 64-71.
- Consorti, F., Merialdo, P., & Sindoni, G. (1996). Metadata reference model for medical documentation: A hypermedia proposal. *Proceedings of the 1st IEEE Metadata Conference*, (<http://www.computer.muni.cz/conferen/meta96/sindoni/ieee.html>).
- Dempsey, L. & Weibel, S.L. (1996). The Warwick metadata workshop: A framework for the deployment of resource description. *D-Lib Magazine*, Jul./Aug. (<http://www.dlib.org/dlib/july96/07weibel.html>), 1996.
- Fluckiger, F. (1995). *Understanding Networked Multimedia: Applications and Technology*, Prentice Hall.
- Frank, U. (1997). Enhancing object-oriented modeling with concepts to integrate electronic documents, *Proceedings of the 30th Hawaii International Conference on System Science*, 6, 127-136.
- Glavitsch, U., Schauble, P., & Wechsler, M. (1994). Metadata for integrating speech documents in a text retrieval system. *ACM SIGMOD Record*, 23(4), 57-63.
- Jain, V & Hampapur, A. (1994). Metadata in video databases. *ACM SIGMOD Record*, 23(4), 27-33.
- Kashyap, V. & Sheth, A. (1996). Semantic heterogeneity in global information systems: The role of metadata, context and ontologies. in *Cooperative Information Systems: Current Trends and Directions*, Papazoglou, M. & Schlageter, G. (Eds.), (<http://lstdis.cs.uga.edu/~kashap/mikep-chapter.ps>).
- Kerherv, B., Pons, A., Bochmann, G.V. & Hafid, A. (1996). Metadata modeling for quality of service management in distributed multimedia systems. *Proceedings of the 1st IEEE Metadata Conference*, (<http://www.computer.muni.cz/conferen/meta96/sindoni/ieee.html>).
- Kiyoki, Y., Kitagawa, T., & Hayama, T. (1994). A metadatabase system for semantic image search by a mathematical model of meaning. *ACM SIGMOD Record*, 23(4), 34-41.
- Lee, H., Kim, J., Kim, Y., & Cho, S. (1999a). A view based hypermedia design methodology. *Journal of Database Management*, 10(2), 3-13.

- Lee, H., Lee, C., & Yoo, C. (1999b). A scenario-based object-oriented methodology for developing hypermedia information systems. *Information and Management*, 36, 34-41.
- Lee, H. & Suh, W. (1999). A workflow-based methodology for developing hypermedia information systems. *Proceedings of the 5th International Conference of the Decision Sciences Institute*, 1, 896-898.
- Meier, J. & Sprague, R. (1996). Towards a better understanding of electronic document management. *Proceedings of the 29th Hawaii International Conference on System Science*, 5, 53-61.
- Mena, E., Kashap, V., Sheth, A., & Illarramendi, A. (1996). OBSERVER: An approach for query processing in global information systems based on interoperability across pre-existing ontologies. *Proceedings of the First IFCIS International Conference on Cooperative Information Systems (CoopIS '96)*.
- Murphy, L.D. (1998). Digital document metadata in Organizations: Roles, analytical, approaches, and future research directions. *Proceedings of the 31th Hawaii International Conference on System Science*, 2.
- Nielsen, J. (1993). *Hypertext and hypermedia*, Academic Press Professional.
- Schwabe, D. & Rossi, G. (1994). *From Domain Models to Hypermedia Applications: An Object-Oriented Approach*, Technical Report MCC 30/94, Dept. of Information, PUC-Rio.
- Shklar, L., Sheth, A., Kashyap, V., & Shah, K. (1995). InfoHarness: Use of automatically generated metadata for search and retrieval of heterogeneous information. *Proceedings of CAiSE 95*.
- Shum, S.B. (1997). Negotiating the construction and reconstruction of organizational memories. *Journal of Universal Computer Science*, 3(8), 899-928.
- Sprague, R.H. (1995). Electronic document management: Challenges and opportunities for information systems managers. *MIS Quarterly*, Mar., 29-49.
- Stein, E.W. & Zwass, V. (1995). Actualizing organizational memory with information systems, *Information Systems Research*, 6(2), 85-117.
- Sutton, M.J.D. (1996). *Document Management for the Enterprise - Principles, Techniques, and Applications*, Wiley.
- Uijlenbroek, J.J.M. & Sol, H.G. (1997). Document based process improvement in the public sector: settling for the second best is the best you can do. *Proceedings of the 30th Hawaii International Conference on System Science*, 6, 107-117.
- Weibel, S., Godby, J., & Miller, E. (1995). OCLC/NCSA metadata workshop report. ([http:// www.oclc.org:5040/oclc/research/metadata/dublin_core_report.html](http://www.oclc.org:5040/oclc/research/metadata/dublin_core_report.html)).

- Weibel, S. & Iannella, R. (1997). The 4th Dublin core metadata workshop report. *D-Lib Magazine*, Jun., (<http://www.dlib.org/jun97/metadata/06weibel.html>).
- Wijnhoven, F. (1998). Designing organizational memories: Concept and method. *Journal of Organizational Computing and Electronic Commerce*, 8(1), 29-55.

Chapter 6

An Adaptive Probe-Based Technique to Optimize Join Queries in Distributed Internet Databases

Latifur Khan

University of Texas at Dallas, USA

Dennis McLeod and Cyrus Shahabi

University of Southern California, Los Angeles, USA

An adaptive probe-based optimization technique is developed and demonstrated in the context of an Internet-based distributed database environment. More and more common are database systems, which are distributed across servers communicating via the Internet where a query at a given site might require data from remote sites. Optimizing the response time of such queries is a challenging task due to the unpredictability of server performance and network traffic at the time of data shipment; this may result in the selection of an expensive query plan using a static query optimizer. We constructed an experimental setup consisting of two servers running the same DBMS connected via the Internet. Concentrating on join queries, we demonstrate how a static query optimizer might choose an expensive plan by mistake. This is due to the lack of a priori knowledge of the run-time environment, inaccurate statistical assumptions in size estimation, and neglecting the cost of remote method invocation. These shortcomings are addressed collectively by proposing a probing mechanism. Furthermore, we extend our mechanism with an adaptive technique that detects sub-optimality

of a plan during query execution and attempts to switch to the cheapest plan while avoiding redundant work and imposing little overhead. An implementation of our run-time optimization technique for join queries was constructed in the Java language and incorporated into an experimental setup. The results demonstrate the superiority of our probe-based optimization over a static optimization.

A distributed database is a collection of partially independent databases that share a common schema, and coordinates processing of non-local transactions. Processors communicate with one another through a communication network (Silberschatz, Korth, and Sudarshan, 1997; Yu and Meng, 1998). We focus on distributed database systems with sites running homogeneous software (i.e., database management system, DBMS) on heterogeneous hardware (e.g., PC and Unix workstations) connected via the Internet. The Internet databases are appropriate for organizations consisting of a number of almost independent sub-organizations such as a University with many departments or a bank with many branches. The idea is to partition data across multiple geographically or administratively distributed sites where each site runs an almost autonomous database system.

In a distributed database system, some queries require the participation of multiple sites, each processing part of the query as well as transferring data back and forth among themselves. Since usually there is more than one plan to execute such a query, it is crucial to obtain the cost of each plan, which highly depends on the amount of participation by each site as well as the amount of data shipment between the sites. Assuming a private/dedicated network and servers, this cost can be computed *a priori* due to the predictability of servers and network conditions and availability of effective network bandwidth. However, in the Internet environment, which is based on a best effort service, there are a number of unpredictable factors that make the cost computation complicated (Paxson and Floyd; 1997). A *static* query optimizer that does not consider the characteristics of the environment or only considers the *a priori* knowledge on the run-time parameters might end up choosing expensive plans due to these unpredictable factors. In the following paragraph, we explain some of these factors via simple examples.

Participating sites (or servers) of Internet database systems might have different processing powers. One site might be a high-end multiprocessor system while the other is a low-end PC running (say) Windows NT. In addition, since most queries are I/O intensive, a site having faster disk drives might observe a better performance. In an Internet-based environment these sites might be dedicated to a single application or multiple simultaneous applications. For example, one site might only run a database server while the other is a database server, a web server,

and an e-mail server. Moreover, the workload on each server might vary over time. A server running overnight backup processes is more loaded at night as compared to a server running 8a.m.-5p.m. office transactions. Due to time differences, a server in New York might receive more queries at 5 a.m. in pacific standard time as compared to those received by a server in Los Angeles. The network traffic is another major factor. It is not easy to predict network delay in the Internet due to variability of effective network bandwidth among the sites. A query plan which results in less tuple shipments might or might not be superior to the one preferring extensive local processing, depending on the network traffic and server load at the time of query processing. Briefly, there is just too much uncertainty and a very dynamic behavior in an Internet-based environment that makes the cost estimation of a plan a very sophisticated task.

Although we believe our probe-based run-time optimization technique is applicable to multi-databases with sites running heterogeneous DBMS, we do not consider such a complex environment in order to focus on the query processing and optimization issues. There has been an extensive research in query processing and optimization in both distributed databases and multi-databases (Amsaleg, Bonnet, Franklin, Tomasic, and Urhan, 1997; Apers, Hevner, and Ya, 1983; Bernstein, Goodman, Wong, Reeve, and Rothnie, 1981; Bodorik, Riordo, and Jacob, 1989; Bodorik, Riordo, and Pyra, 1992; Chen and Yu, 1992; Evrendile, Dogac, Nural, and Ozcan, 1997; Kambayaashi, Yoshikawa, and Yajima, 1983; Roussopoulos and Kang, 1991; Zhu and Larson, 1994). Among those, only a few considered run-time parameters in their optimizers. We distinguish these studies from ours in Related Work section. Briefly, most of these studies propose a detective approach to compensate for lack of run-time information while our approach is first predictive and prevents the selection of expensive queries at run-time and then becomes adaptive to adapt itself with run-time variations. In this paper, we demonstrate the importance and effectiveness of an adaptive probe-based optimization technique for join queries in the Internet databases. We focused on *join* queries because join operation is not only frequently used but also expensive (Yu and Meng, 1998).

In order to demonstrate the importance of run-time optimization, we implemented an experimental distributed database system connected through the Internet. Our setup consists of two identical servers both running the same object-relational DBMS (i.e., Informix Universal Server, (Informix, 1997)) connected via the Internet. We then split the BUCKY database (from the BUCKY benchmark (Carey et al., 1997)) across the two sites. We implemented a probe-based run-time optimization module for join queries in Java language. The optimizer first issues two probe queries each striving to estimate the cost of either semi-join or simple join plans. Consequently, the cheapest plan will be selected. The query optimizer of a distributed database system can be extended with our probe queries to capture run-

time behavior of the environment. Furthermore, as a byproduct, the result of the probe queries can be utilized for estimating the size of intermediate relations in a join plan. This estimation is shown to be less sensitive to statistical anomalies as compared to that of static optimizers. Finally, the probe-based technique identified some hidden costs (e.g., the cost of remote invocation of methods with RMI) that should be considered in order to select the cheapest plan. That is, our probing mechanism can capture any surprises associated with specific implementations (e.g., RMI in our case) which can never be accounted for by static optimizers. The experiments show that for expensive queries processing many tuples the response time can be improved on the average by 32.5% over a static optimizer while the probing overhead only results in an average of 6.4% increase in response time. We also discuss an enhanced version of our optimizer, which reduces the overhead by an average of 45% (i.e., observing 3.5% increase in response time due to overhead) by utilizing the results of the probe query. Obviously, these numbers depend on the number of tuples sampled by the probe queries and the size of relations. In addition, we propose an adaptive optimization technique that copes with sudden changes of run-time environment on the fly during the execution of query. However, we show that adaptive optimization incurs either no overhead or a little overhead (only in a few cases).

The remainder of this paper is organized as follows. Related Work section covers some related work on query processing and optimization in both distributed databases and multidatabases. Run-Time Optimization for Join Queries section states the problem, reviews a conventional solution, and finally explains our proposed extensions to capture run-time parameters and utilize them to improve the optimizer. Performance Evaluation section consists of a performance study to compare the performance of our run-time optimization technique with that of a static optimizer. Finally, Conclusions and Future Directions section concludes the paper and provides an overview on our future plans.

RELATED WORK

There have been various studies on query processing and optimization in distributed, federated, and multidatabase systems (Apers, Hevner, and Ya., 1983; Bernstein et al. 1981; Bodorik, Pyra, and Riordo, 1990; Chen and Yu, 1992; Kambayaashi, Yoshikawa, and Yajima, 1983; Roussopoulos and Kang, 1991). What distinguishes us from these studies is our consideration of run-time environment in order to optimize the queries. There are, however, other studies considering run-time environment (Amsaleg, Bonnet, Franklin, Tomasic, and Urhan, 1997; Urhan, Franklin, and Amsaleg, 1998; Bodorik, Pyra, and Riordo, 1989; Bodorik, Pyra, and Riordo, 1992; Ozcan, Nural, Koskal, Evrendile, and Dogac, 1997; Cole

and Graefe, 1994; Zhu and Larson, 1994; Antoshenkov, 1993). Here we discuss those studies in more details and distinguish them from this study.

In Cole and Graefe (1994), they proposed a dynamic query optimization model in a *centralized* database system in order to solve the problem of unknown run-time bindings for host variables in embedded queries. In Antoshenkov (1993), several plans in a centralized database system are executed simultaneously for a short time, and finally, all plans but the best are terminated. The purpose of these simultaneous runs is to capture the cost function instability for a single table access. However, in a distributed database system, these simultaneous runs at a site will compete with each other over resources and they do not correctly capture the run-time environment.

In Evrendile et al. (1997) and Ozcan et al. (1997) assuming a multi-database system they realized the importance of run-time optimization (they used the term *dynamic* optimization) and proposed a weight function to capture the workload and transmission cost for each participating site. The objective is to choose the sites whose cost functions are less than a certain threshold in order to participate them in the query execution. They use a similar technique to our probing mechanism to capture the workload; however, the communication cost is calculated as the proportionate to the size of the tuples transmitted. Due to network bandwidth variability over the Internet, it is not possible to estimate the communication cost *a priori*. In addition, among different sites over the Internet, network bandwidth may vary significantly. Finally, we show that other factors such as server load and choice of implementation also impact the communication cost.

In Zhu and Larson (1994), a query sampling technique was proposed to estimate the cost parameters of an autonomous local database system in order to perform global query optimization in a multi-database system. Their objective is to estimate the local costs offline in order to later utilize them by a global query optimizer to determine good execution plans for a series of queries. Therefore, the overheads of sample queries are not important. In addition, their approach does not address when and how the sampling should be invoked to capture a changing environment at run-time. However, our probing mechanism cannot afford to do a complex statistical analysis on samples because it is invoked per query and hence needs to impose a low overhead on the system.

In Amsaleg et al. (1997) and Urhan et al. (1998) they also realized the inadequacy of static query optimization and proposed a *detective* technique to identify sites with delays higher than expected during query execution. Subsequently, they stop waiting for problematic sites and reschedule the plan for other sites in order to hide delays by pushing the delayed sites as far back in the optimizer plan as possible. In this case, their technique might generate incomplete results if the problematic sites never recover. While their approach detects the problem and try

to resolve it, our approach is *predictive* and tries to avoid the problem all together. Although a predictive approach results in an initial overhead, we show that in some cases we can minimize the overhead by utilizing the results of our probing query. Furthermore, for expensive queries the overhead is marginal. Similar to previous studies, in their simulation model they assume communication cost is proportional to the size of data transfer in bytes.

In Bodorik et al. (1989) various types of adaptive query execution techniques are discussed. The idea is to monitor the execution of a plan and if the performance is lower than what estimated, then the plan is corrected utilizing the newly captured information. Again, this is a detective technique trying to compensate for a wrong decision by re-planning. Finally, Bodorik et al. also assume communication costs are directly proportional to the volume of data transferred.

In Mackert and Lohman (1986), query optimizer estimates the total cost of a plan by summing up the CPU cost, I/O cost, message passing cost and communication cost. The last two costs are computed based on heuristics. Communication cost is estimated as the product of number of bytes transferred and effective bandwidth available between the two sites. Over the Internet, it is not trivial to obtain the effective bandwidth between two sites. Furthermore, effective bandwidth is changing frequently due to the network dynamics and it is hard to maintain updated effective bandwidth information.

RUN-TIME OPTIMIZATION (RTO) FOR JOIN QUERIES

In this section, we start by defining the problem of query optimization for join queries in distributed databases. Subsequently, we briefly describe a conventional solution to the problem. Finally, we propose our probing mechanism and compare it with the conventional solution. Note that query optimization within a database site is beyond the scope of this paper and our techniques rely on each site for local query optimizations.

Problem Statement

Suppose there are two relations R_l at local site S_1 and R_r at remote site S_2 . Consider the query that joins R_l and R_r on attribute A and requires the final result to be at S_1 . The objective is to minimize the query response time. A straightforward plan, termed *simple join plan* (P_j), is to send relation R_r to site S_1 and perform a local join at S_1 . This approach observes one data transfer and one join operation. The second plan employs semi-join and is termed *semi-join plan* (P_{sj}). This strategy incurs two data transfers and also performs join twice. Utilization of semi-joins to reduce the size of the intermediate relations has received a great deal of

attention (Bernstein et al., 1981; Yu and Meng, 1998). The decision between choosing one plan over the other is not straightforward and depends on a number of parameters such as the size and cardinality of relations R_l and R_r . Therefore, the problem is how to decide which plan to choose in order to minimize the response time of a certain join query. It is the responsibility of a query optimizer to assign a cost to each plan and then choose the cheaper plan. Intuitively, if $|R_l|$ and $|R_r|$ are the cardinality of relation, R_l and R_r , respectively, then when $|R_l| \ll |R_r|$, the semi-join plan seems promising and vice-versa.

Static Query Optimizer (SQQ)

In this section, we explain a conventional method (Bernstein et al. (1981; Ceri and G. Pelagatti, 1984; Apers, Hevner, and Ya, 1983) to estimate the costs associated with both simple join and semi-join plans. Since the parameters used for this cost estimation are all known a priori before the execution of the plans, this query optimizer is termed *Static Query Optimizer (SQQ)*.

Given the number of tuples in R_r as N_r and the size of a tuple in R_r as S_{Rr} , the cost of simple join is trivially computed as follows:

$$Cost(P_j) = C_0 + C_1 \times S_{Rr} \times N_r \quad (1)$$

where C_0 is the cost to startup a new connection and C_1 is the communication cost per byte transfer.

The computation of the cost of semi-join plan is more complicated:

- a. Let us denote the size of the common attribute A as SR_A , and the number of distinct values for attribute A in local relation (R_l) as N . The cost to transfer $\Pi_A(R_l)$ from S_1 to S_2 is:

$$C_0 + C_1 \times S_{RA} \times N_\ell \quad (2)$$

- b. Now $\Pi_A(R_l)$ is joined with R_r at S_2 with a zero cost ($R' = \Pi_A(R_l) \bowtie R_r$)

- c. Suppose $|R'|$ is the cardinality of relation R' , the cost of sending R' to S_1 is:

$$C_0 + C_1 \times S_{Rr} \times |R'| \quad (3)$$

- d. Finally, R' is joined with R_l at S_1 with a zero cost ($Res = R_l \bowtie R'$)

Therefore, the overall cost for the semi-join plan is

$$Cost(P_{sj}) = 2 \times C_0 + C_1 \times (S_{RA} \times N_\ell + S_{Rr} \times |R'|) \quad (4)$$

SQO will choose the semi-join plan if $\text{Cost}(P_{sj}) \leq \text{Cost}(P_j)$, or if:

$$C_0 + C_1 \times (S_{RA} \times N_\ell + S_{Rr} \times |R'|) \leq C_1 \times S_{Rr} \times N_r \quad (5)$$

SQO can examine the above inequality accurately only if it has all the required information (e.g., N_p, S_{Rr}) a priori. Note that C_1 is simply reciprocal of network bandwidth. However, over the Internet, effective network bandwidth between two sites is extremely difficult to estimate because it is changing more frequently. Finally, SQO needs to estimate the size of intermediate results (i.e., $|R'|$). One estimation is as follows:

$$|R'| = \text{domain}(A) \times \text{sel}(R_\ell, A) \times \text{sel}(R_r, A) \quad (6)$$

where $\text{sel}(R_\ell, A)$ and $\text{sel}(R_r, A)$ are the selectivity of attribute A in relations R_ℓ and R_r , respectively. Eq. 6 is based on two assumptions. First, it assumes that the domain of A is discrete and can be considered as A 's sample space. Second, tuples are distributed between R_ℓ and R_r independent of the values of A . That is, there is no correlation between R_ℓ and R_r based on the join attribute A . Later in the following paragraphs, we show that as a by product of our probing technique, we do not need to make any of these assumptions.

Our Proposed Solution

We now describe our *run-time optimization (RTO)* technique, which is an extension to SQO. To summarize, RTO first submits two *probe* queries to estimate the run-time costs corresponding to plans P_j and P_{sj} by measuring the response time observed by each probe query. Subsequently, it replaces $C_1 \times S_{RA}$ and $C_1 \times S_{Rr}$ in Eq. 5 by the estimated costs. In addition, RTO analyzes the results of the probe queries to estimate the size of R more accurately. This last step of RTO is of course identical to the concept of sampling. For the time being, we assume that there would be no sudden changes in the behavior of run-time environment between the time that a probe query is submitted and the time that the original query will be executed. This assumption is relaxed in later in the discussion.

For the remainder of this section, we first describe the probe queries and how their measured performance values are incorporated into Eq. 5. Next, we propose an enhanced version of RTO to reduce the overhead of probing by utilizing its results to support the original query. Later, in the following paragraphs, we argue how our modification to Eq. 5 can capture run-time behavior and estimate the size of intermediate relations more accurately. Finally, we show how our optimizer copes with sudden changes of run-time environment.

Probe Queries

Our main objective is to modify the SQO main equation (Eq. 5) in order to take the run-time parameters into the consideration. To achieve this, we submit the following two probe queries to collect some parameters at run-time:

Probe Query A: The first probe query strives to replace the term $C_1 \times S_{RA}$ of Eq. 5 with a more accurate estimation. This is because $C_1 \times S_{RA}$ is based on the simplistic assumption that communication cost is a linear function of the amount of data transferred and network bandwidth ($1/C_1$) is also available. This probe sends the A attribute of X number of tuples of R_l , denoted as R_{XA} , from local site S_1 to remote site S_2 ; joins R_{XA} with R_r at remote site S_2 ; and receives back the size of the result denoted as X_j . The time to execute this probe query is measured and then is normalized by dividing it by X. The result is the cost of this probe and is denoted by C_{12r} . To illustrate the costs that have been captured by C_{12r} , consider the following equation:

$$C_{12r} = \frac{S(X) + RIC_Q + RIC(X) + JC_r}{X} \quad (7)$$

In Eq. 7, $S(X)$ is the cost to ship X tuples (each tuple consists of only one attribute A) from S_1 to S_2 , RIC_Q is the remote invocation cost for the join operation at S_2 , $RIC(X)$ is the remote invocation cost to insert X tuples into S_2 , and JC_r is the cost to perform the join operation at S_2 . Observe that as a byproduct, R' can now be estimated more accurately because X_j is the number of tuples in R' if R_l had X tuples. Now that R_l has N_l tuples then size of R' can be estimated as:

$$Sample_estimate(R') = \frac{X_j \times N_l}{X} \quad (8)$$

Probe Query B: The second probe query strives to replace the term $C_1 \times S_{Rr}$ of Eq. 5 with a more accurate estimation. It receives X number of tuples of R_r , denoted as R_x , from remote site S_2 ; joins R_x with R_l at local site S_1 ; and measures the time to complete this process. This time is then normalized by dividing it by X and is the cost of this probe (denoted by C_{r2l}). To illustrate the costs that have been captured by C_{r2l} , consider the following equation:

$$C_{r2l} = \frac{RIC(X) + S(X) + JC_l}{X} \quad (9)$$

In Eq. 9, $S(X)$ is the cost to ship X tuples from S_2 to S_1 , JC_l is the cost to perform the join operation at S_1 , and $RIC(X)$ is the remote invocation cost to

request X tuples from S_2 . In Eqs. 7 and 9, $S(X)$ is capturing the following run-time parameters:

$$S(X) = Delay_{send}(X) + Delay_{network}(X) + Delay_{receive}(X) \quad (10)$$

where $Delay_{send}(X)$ is the time required at the sender site to emit X tuples, $Delay_{receive}(X)$ is the time required at the receiver site to receive X tuples and $Delay_{network}(X)$ is the network delay. It is important to note that shipment cost, remote invocation cost, and join cost are intermixed in C_{l2r} and C_{r2l} . This is not an obstacle in our case since it is not required to estimate each of these costs separately.

Now we can modify Eq. 5 of SQO as follows:

$$N_\ell \times C_{l2r} + Sample_estimate(R') \times C_{r2\ell} \leq N_r \times C_{r2\ell} \quad (11)$$

In Eq. 11, the terms $C_1 \times SR_A$, and $C_1 \times SR_r$ of Eq. 5 are replaced by C_{l2r} and C_{r2l} ; and R' is computed using Eq. 8 instead of Eq. 6.

Selection of X tuples: Both probe queries transfer X tuples for their estimations. Therefore, the value of X (i.e., the number of tuples transferred) has an impact on the accuracy of the estimations. Trivially, the larger the value of X the more accurate the estimation. Moreover, the amount of data transferred for X should be large enough to exercise the network's TCP connection beyond its slow start. However, large value of X results in more overhead observed by the probe queries. In our experiments, we varied X from 1% to 10% of N_l . Besides the value of X , the way that X tuples are selected impacts the estimated size of R' . This sampling should be done in a way that X be a good representative of R_l . This can be achieved by random selection of tuples from the relation R_l . There are alternative techniques described in the literature for random selections of tuples from a relation such as heap scan, index scan and an index sampling technique (Olken, 1993; Peter and Arun, 1995). There are many issues in obtaining a good random representative specially when there are index structures on the relation. The details of sampling are beyond the scope of this paper.

Scalability: Although we describe our probe queries for joins between two relations (i.e., 2-way join), the technique is indeed generalizable to k -way join. When joining k relations on a common attribute, the k -way join can be considered as $(k-1)$ 2-way joins. The purpose of this join is to reduce the size of relations and determine which tuples of relations are participating in the final result. Finally, all processed relations are transmitted to a final site where joins are performed and the answer to the query obtained (Chen, and Victor, 1984). Hence, the optimization challenge in the reducing phase is to identify the optimal execution order of the k -

way join. Static optimizers for distributed databases address this challenge by sorting the k relations in ascending order of their volumes (Ayers, Hevner, and Ya, 1983). Assuming the communication cost is independent of the network load and is linearly proportional to the volume of transferred data, then this sorted order specifies the optimal execution order. But over the Internet, network bandwidth among the sites vary significantly. This variable network load should be taken into account to identify the optimal plan. Therefore, our probe-based technique can be utilized in a similar way to estimate the communication cost among all the k participating sites (assuming one relation per site). As a result $k \times (k-1)$ probe queries are generated among the k sites. One can argue that our technique is not scalable due to the extensive increase in the number of probe queries in a k -way join optimization. However, it is important to note that these probe queries are independent of each other and thus can be executed in parallel. In our experiments, we utilized Java multithreading primitives (Reese, 1997) to perform probe queries concurrently. Therefore, the overhead observed for k -way join optimization is equal to the maximum delay incurred among all the probe queries. After estimating the communication costs from site to site, the optimal execution order is determined by the ascending order of number of tuples transferred multiplied by the communication cost between the corresponding sites. It is important to note that our probing technique does not require to know the network bandwidth among the sites. On the fly, it inherently captures the network bandwidth and takes into consideration sites' loads and remote invocation costs. Currently, we are investigating the extension of our probe-based technique to support k -way join within our experimental setup.

Enhanced RTO

One major problem with our RTO is the overhead associated with probing queries. This overhead can be alleviated by a simple enhancement. Recall that during the first step of probe query A , X tuples of R_1 each consisting of single common attribute A are transferred to S_2 . The idea is to keep that relation R_{XA} at S_2 and do not discard it. Therefore, if P_{sj} is selected by RTO as the superior plan, it will not be required to send that X tuples to S_2 again. This results in saving both $S(X)$ and $RIC(X)$. We evaluated the impact of this enhancement in our performance evaluation and an average of 45% reduction in overhead has been observed for a given value of X .

Analysis and Comparison

In this section, we analyze why Eq. 11 can now capture run-time behavior and estimate the size of intermediate relations more accurately than SGO.

Communication Cost

Almost all the previous studies on distributed query optimization (see Related Work section) assumed communication cost is proportional to the size of data transferred. They also assume network bandwidth information is available to the system and remains constant. This is reasonable for a private/dedicated network. The same assumptions have also been made by the static query optimization technique discussed in this paper (see Static Query Optimizer section). However, researchers (Paxson, 1997) in network community demonstrate that over the Internet, it is hard to estimate the effective network bandwidth. In addition, network bandwidth between two sites varies significantly with time due to the Internet dynamics. In our experiments, however, we observed that the communication cost is indeed a linear function of the number of tuples transferred. This is because the granularity of data transfer in our experiments was in tuples. With RTO , C_{l2r} and C_{r2l} are the linear extrapolation of the time to move X tuples and hence are based on number of tuples moved at the time of query execution between the two participating sites. In addition, the size of tuples is also taken into consideration by measuring the actual time to transfer X tuples of size S_{RA} and S_{Rr} . By doing this, we are inherently capturing the available network bandwidth between two sites at run-time. Note that the same argument holds if the granularity of data transfer is in blocks instead of tuples. However, the probe queries must be modified to extrapolate on the number of block movement as opposed to tuple movement. This is a straightforward extension.

Remote Invocation Cost

As discussed in later sections, in our experimental setup Remote Method Invocation (RMI) was employed in order to access a remote server. An interesting distinction between simple join and semi-join plan is that in general semi-join plan uses remote invocation more often as compared to that of simple join plan. To illustrate, P_{sj} utilizes remote invocation N_1 times to insert tuples into S_2 , one time to execute join remotely at S_2 , and R' times to fetch the semi-join results back to S_1 . This is while P_j utilizes RMI only N_r times to fetch the remote tuples into S_1 . Obviously, this hidden RMI cost has not been captured by SQO because this cost is very specific to our implementation and experimental setup. The interesting observation, however, is that this cost has automatically been captured by C_{r2l} and C_{l2r} . Therefore, a general conclusion is that our run-time probing mechanism can capture any surprises associated with specific implementations (e.g., RMI in our case) which can never be accounted for by the static optimizer. Note that other alternative implementations will also observe some overheads similar to that of RMI. For example, if Java Database Connectivity (JDBC) is employed to connect to the database servers, remote sites can be accessed in three alternative ways

depending on the JDBC driver implementation (Reese, 1997): 1) distributed objects implemented in RMI, 2) message passing technique, or 3) Common Object Request Broker Adapter (CORBA) (Farley, 1998). Trivially, all three methods introduce some overheads when accessing remote sites. Hence, C_{r2l} and C_{l2r} automatically capture these varying overheads regardless of different implementations of JDBC.

Load Cost:

From Eq. 5, it is obvious that SQO does not consider the time to process different operations such as project, join and semi-join which are impacted by server workload. This is because it assumes that communication cost is the dominant factor in estimating the cost of a plan. However, load has an important impact in choosing the best plan. On the other hand, with RTO, it is trivial from Eqs. 7, 9, and 10 that the workload of the server can be captured by C_{r2l} and C_{l2r} due to the following terms: JC_r , JC_l , $\text{Delay}_{\text{send}}$, and $\text{Delay}_{\text{receive}}$. Hence, another distinction between P_{sj} and P_j can be captured by our RTO. That is, semi-join performs two light joins one at remote site and the other at local, while simple join only performs one heavy but local join operation. Beside these operations that are highly dependent on the server workload, there are other dependencies as well. A heavily loaded server also impacts the communication cost since it sends and receives tuples slower than a lightly loaded server (i.e., $\text{Delay}_{\text{send}}$ and $\text{Delay}_{\text{receive}}$). Consequently, it is not straightforward to model the impact of load on the cost of a plan. This is exactly why our probing mechanism can automatically capture these chaotic behaviors and aggregate them out within two simple terms of C_{r2l} and C_{l2r} .

Statistical Assumptions

Regarding the statistical assumptions, RTO has two major advantages over SQO. First, RTO does not rely on remote profiles. Accessing metadata from the remote sites is not easy because statistic profiles are changed frequently and hence the process of collecting and updating the statistical information about the remote site is expensive. Recall that while SQO needs the value of S_{R_r} and $\text{sel}(R_r, A)$ for its computations, RTO relies on neither of these values. Second, RTO is less sensitive to the statistical anomalies as compared to SQO. SQO makes two major assumptions in order to estimate the size of R' in Eq. 6: 1) domain of A is discrete and can be considered as A 's sample space, and 2) there is no correlation between R_l and R_r . Instead, RTO estimates the size of R' by sampling (see Eq. 8) and thus is independent of both of these assumptions. That is, with RTO, A 's sample space is R_p ; moreover, it utilizes the entire R_r which is R_r 's best possible sample. In addition, if there is a correlation between the two relations, it will impact X_j (in Eq. 8) accordingly. Therefore, a positive correlation results in higher value of X_j and vice-versa.

An Adaptive Optimization Technique

We made the simplifying assumption that there would be no sudden changes in the runtime environment between the time the probe queries are submitted and the time the original query is executed. However, in some cases, a single probing may not be enough to predict the run-time environment during the original query execution time. This is because some queries might take minutes to execute and hence there is a possibility of changes in the run-time parameters. Therefore, it is necessary to examine during the query execution whether the selected plan still provides optimal solution or not. If not, then the optimizer should discard the plan and choose a new one. Moreover, the new plan should be intelligent enough to avoid redundant work that has already been done by the earlier plans.

With our *adaptive optimization* technique, we partition a join query into K series of smaller joins. Subsequently, for each smaller join, we re-evaluate the run-time parameters and make a decision to either continue with the current plan or switch to another plan. Our technique, however, does not treat each smaller join in isolation. It ensures that no smaller join performs redundant work that has already been done by the previous joins in the series. Briefly, the optimizer collects statistics to update the cost model at each re-evaluation point, *termed cost-update points*. Using the updated cost model, costs of different plans to complete the query are estimated and the optimizer chooses the least expensive one. To achieve this, we need to recompute C_{l2r} and C_{r2l} at each cost-update point. For most of the cases, our adaptive technique can estimate C_{l2r} and C_{r2l} by just timing the execution of plan as it progresses. Hence, new probe queries are not required to be sent explicitly. For other cases, it needs to submit new probe queries. The number of probe queries submitted explicitly for K series of joins is shown in Table 1. These extra probe queries can either be issued at the *cost-update point* or being executed on the background during the execution of the plan. Both approaches have advantages and disadvantages. The former observes an overhead for cases where the next selected plan is not P_{sj} (assuming our enhanced RTO). The latter does not observe this overhead but it may overestimate the parameters because itself might overload the system. To explain how our technique decides on a plan for each smaller joins

Table 1: Probe query overheads for adaptive optimization.

Plan	Number of Probe	
	Query A	Query B
P_{sj} plan observed for all K cost-update points	1	1
P_j plan observed for all K cost-update points	K	1
P_j and P_{sj} plan observed alternatively	1	1

and how it avoids redundant work in case of a switch of plans, we need to define some terms.

Definition 1: Let there be K cost-update points, cu_1, cu_2, \dots, cu_k , then a plan, P_{cu_i} is selected by the optimizer at cu_i where $P_{cu_i} \in \{P_j, P_{sj}\}$.

Definition 2: If $N(P_{cu_i})$ tuples are transmitted from one site to another at cu_i then for $P_{cu_i} = P_j$, $N(P_{cu_i})$ tuples of R_r relation are sent from S_2 to S_1 . However, if P_{sj} is selected at cu_i , then $N(P_{cu_i})$ tuples of R_l relation over common attribute A are sent from S_1 to S_2 .

In order to avoid redundant tuple transfers, $N(P_{cu_i})$ tuples are chosen in plan P_{cu_i} such that none of these tuples have been transmitted before from S_1 to S_2 by P_{cu_m} where $1 \leq m < i$ and $P_{cu_i} = P_{sj}$. Let $P_{cu_{i-1}} = P_{sj}$, and P_{cu_i} be the plans at cu_{i-1} and cu_i , respectively, then for $P_{cu_i} = P_{sj}$, the number of tuples of R_l that are required to transfer from S_1 to S_2 is:

$$N_l - \sum_{m=1}^{i-1} N(P_{cu_m}) \quad (12)$$

These tuples are joined with R_r at S_2 and finally, the expected number of tuples that are further required to ship from S_2 to S_1 is:

$$\left(1 - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_l} - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_r} \right) \times \text{Sample_estimate}(R') \quad (13)$$

The subtracted terms presents the expected number of tuples that have already been transferred. At cu_i , C_{l2r} and C_{r2l} are updated with the recent P_{sj} cost estimate for $P_{cu_{i-1}} = P_{sj}$. Note that for the recent P_{sj} cost estimate,

$$\frac{N(P_{cu_{i-1}})}{N_l} \times \text{Sample_estimate}(R') \text{ tuples were expected to ship from } S_2 \text{ to } S_1.$$

This fact is taken into account during the estimation of C_{r2l} . Hence, the overall cost for the P_{sj} plan to perform join for the rest of the tuples at cu_i is:

$$\begin{aligned} \text{Cost}(P_{sj}) = & (N_l - \sum_{m=1}^{i-1} N(P_{cu_m}) \times C_{l2r} \\ & + \left(1 - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_l} - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_r} \right) \times \text{Sample_estimate}(R') \times C_{r2l} \end{aligned} \quad (14)$$

Let $P_{cu_{i-1}} \in \{P_j, P_{sj}\}$, and P_{cu_i} be the plans at cu_{i-1} and cu_i respectively, then for $P_{cu_i} = P_j$, the number of tuples of R_r that are required to transfer from S_2 to S_1 is

$$N_r - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_i} \times \text{Sample_estimate}(R') - \sum_{m=1}^{i-1} N(P_{cu_m}) \quad (15)$$

Therefore, the cost of the P_j plan to perform join for the rest of the tuples at cu_i is

$$\text{Cost}(P_j) = (N_r - \frac{\sum_{m=1}^{i-1} N(P_{cu_m})}{N_i} \times \text{Sample_estimate}(R') - \sum_{m=1}^{i-1} N(P_{cu_m})) \times C_{r2l} \quad (16)$$

Note that if $P_{cu_{i-1}} = P_j$, C_{r2l} is updated at cu_i . In this case, C_{l2r} cannot be estimated unless either a new probe query A is issued at cu_i or probe query A has already been issued in the background during the execution of P_j . The overhead of probe query A can be entirely avoided if $P_{cu_i} = P_{sj}$. Finally, if $P_{cu_{i-1}} = P_{sj}$, C_{r2l} and C_{l2r} are updated at cu_i accordingly and no extra probe query is required. Hence, RQO with adaptive optimization chooses P_{sj} plan if $\text{Cost}(P_{sj}) \leq \text{Cost}(P_j)$; otherwise, the optimizer switches to P_j . It is important to realize that once a tuple of R_r is sent by a certain plan P_{cu_m} from S_2 to S_1 , that tuple is not sent again even if it is selected in a plan P_{cu_i} where $m < i$, and $P_{cu_i} \in (P_{sj}, P_j)$ and $P_{cu_m} \in (P_{sj}, P_j)$. Therefore, in a plan P_{cu_i} , we select tuples from R_r which were not sent to S_1 in P_{cu_m} , $1 \leq m < i$. In order to do this, as SQL operation will be executed remotely, hence JC_r now becomes expensive due to the additional condition (see Eq. 7). Finally, for each P_{cu_i} , after gathering tuples from S_2 , final join is carried out at S_1 between the R_i and the shipped data set.

There is a trade-off in determining the frequency of *cost-update points*. Checking too many points for cost-update can lead to an unacceptably high overhead. In contrast, *few cost-update points* may result in loss of some optimization opportunities. For K cost-update points, the overhead for probe queries A and B are depicted in Table 1. Trivially, K is a function of $N(P_{cu_i})$, number of plan switches and their execution orders. For now, we assume equal values of $N(P_{cu_i})$ for different cu_i and we fix $N(P_{cu_i})$ at X . However, we are investigating how to choose K in order to strike a compromise between these trade-offs in order to impose a minimum overhead on the system.

PERFORMANCE EVALUATION

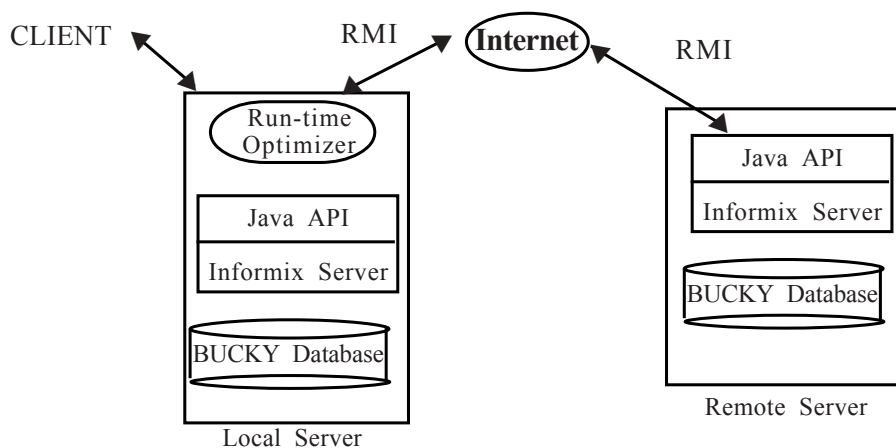
As we argued in earlier, the run-time behavior is too unpredictable and sophisticated to be captured and analyzed by analytical models or simulations. Hence, we decided to implement a real experimental setup. We conducted a number of experiments to demonstrate the superiority of RTO over SQO for join queries. In these experiments, first we varied the workload on the two servers in order to simulate a heterogeneous environment and/or variable run-time behavior of the environment. Our experiments verified that RTO can adapt itself to workload

changes and always chooses the best plan while SQO's decision is static and always a specific plan is chosen independent of the load on the servers. Second, our experiments showed that even in case of a balance load, RTO outperforms SQO because it captures both the communication cost and the overhead attributed to a specific implementation setup (e.g., RMI cost) correctly. We did not report our experiments for variable network load because both of our join plans utilize network almost identically and hence a congested (or free) network will not result in preferring one plan to the other. We plan to do more experiments with other sorts of queries that give rise to plans utilizing network differently. Finally, we did some experiments to investigate the overhead associated with probe queries and quantify the reduction in overhead by employing our enhanced version of RTO. In these experiments, we did not vary the run-time environment during the query execution. Therefore, more experiments are required to study the effectiveness of our adaptive optimization technique.

Experimental Setup

Figure 1 depicts our experimental setup which consists of two sites S_1 and S_2 which are not within a LAN but within the campus area network (CAN). The sites are Unix-boxes with an identical hardware platform (a SUN Sparc Ultra 2 model with 188 MBytes of main memory and 100 clock ticks/second speed). The buffer pool was kept at 0.4 MB for the system. We intentionally chose not to have a large buffer pool to avoid the database becomes memory resident. This is because we wanted to study the effect of load over communication cost. Note that in our experiments we degrade the performance of one server by loading it with additional processes and emulating an environment with heterogeneous servers. Each process increases server disk I/O by repeatedly running Unix "find" system call. The additional load is quantified by the number of these processes spawned on a server.

Figure 1: *Experimental Setup*



Each site runs an Informix Universal Server (IUS) which is an object-relational DBMS. The run-time optimizer and its different plans are implemented in Java. The run-time optimizer communicates with the database servers through Java API which is a library of Java classes provided by Informix. It provides access to the database and methods for issuing queries and retrieving results. From applications running on one site, Remote Method Invocation (RMI) is used to open a connection to the database server residing on the other site. The *Credential* class of RMI has a public constructor that specifies enough information to open a connection to a database server. Two types of Credentials are used: 1) *Direct Credentials* for local applications, and Remote Credentials to access the remote database server using typical HTTP credentials. The BUCKY database, from the BUCKY benchmark (Carey et al. 1997), was distributed across the two sites.

The queries are submitted to site S_1 as a local server and might require data to be shipped from site S_2 which is the remote server. RTO resides at S_1 and employs RMI and its HTTP credentials to access the remote site. We concentrate on the two *TA* and *PROFESSOR* relations of BUCKY. The *TA* relation (or R_1) at S_1 and the *PROFESSOR* relation (or R_2) resides at S_2 . For example, in a real-world university application, the information on faculty is kept at a site in the human resources (S_2) while the *TA* information is kept at (say) computer science department site (S_1). The number of tuples per relation residing on each site has been varied for our experiments. We fixed the total number of tuples (i.e., $N_1 + N_2$) at 25,000. Without loss of generality and to simplify the experiments we assumed no duplications in the relations.

The join query is: Find the Name, Street, City, State, Zipcode for every *TA* and his/her advisor, in SQL:

```
Select T.Name, T.Street, T.City, T.State, T.Zipcode,
P.Name, P.Street, P.City, P.State, P.Zipcode
from TA T, PROFESSOR P
where T.advisor=P.id
```

The size of the join attribute id/advisor (i.e., S_{RA}) is 4 bytes and the size of attributes Name, Street, City, State, and Zipcode of *PROFESSOR* relation are 20, 20, 10, 20 and 6 bytes, respectively. When the query is submitted through an interface (a Java applet running at S_1), the query optimizer consults the metadata to identify the location of the *TA* and the *PROFESSOR* relations. RTO will then decide using *probe queries* A & B which plan to choose. We varied the number of tuples per relation (the X-axis of the reported graphs) and measured the response time of the join query (in milliseconds) for each tuple distribution (the Y-axis of the reported graphs). The X-axis is the percentage of the number of tuples of *TA* relation that

resides at S_1 (i.e., $100 \times \frac{N_\ell}{N_\ell + N_r}$). For comparison purposes, we also measured the response time of SQO, *semi-join* and *simple join* for each experiment.

Results

For the first set of experiments, we compared the performance of SQO and RTO when the two servers are equally loaded. In this case, one expect to see a similar performance for SQO and RTO. However, as seen in Fig. 2, RTO (the dotted line) always chooses the correct plan by switching from semi-join to simple join plan at 50% tuple distribution. Instead, SQO (the solid line) wrongly continues preferring semi-join to simple join until 80% of tuple distribution. That is, when the difference between N_ℓ and N_r is significant, both optimizers can correctly determine the best plan. The decision becomes more challenging when N_ℓ and N_r have values with marginal differences. SQO prefers semi-join because it overestimates the communication cost of simple join due to Eq. 5. RTO, however, realizes that communication cost is not only affected by the amount of data shipped but also other factors and hence simple join which ships more volume of data might not be as bad as expected. In this situation, the cost of remote invocation impacts semi-join more than simple join. By capturing the facts and amortizing the associated cost by incorporating C_{12r} and C_{r2l} into its equations, RTO detects the superiority of simple join to semi-join after 50% tuple distribution. Note that switching at the point of 50% tuple distribution cannot be generalized by a static optimizer because it is very much dependent on our experimental setup and the participating BUCKY relations. This is exactly why a run-time optimizer is required.

Figure 2: Response Time of a Join Query for Different Plans

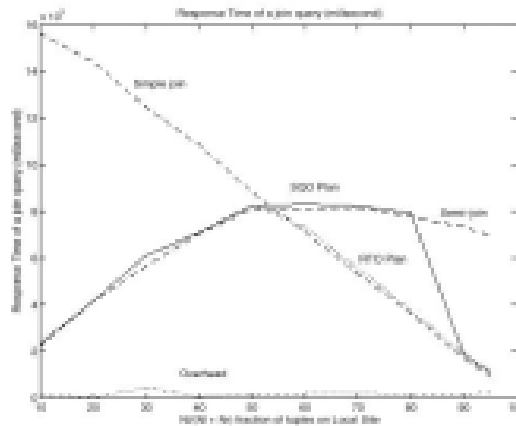
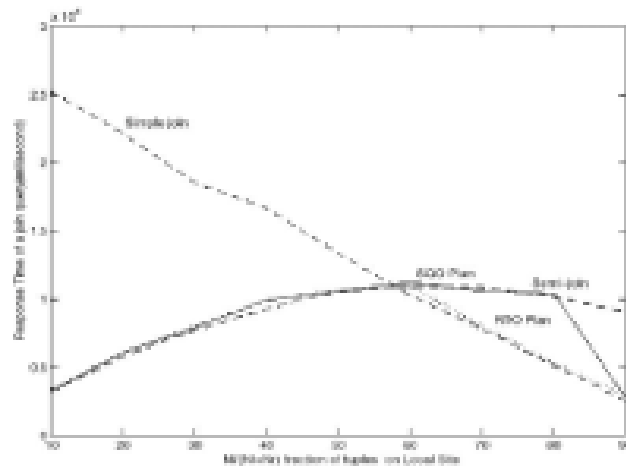
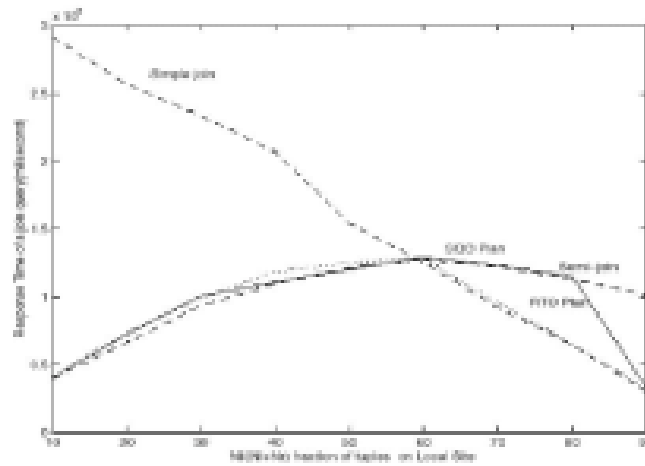
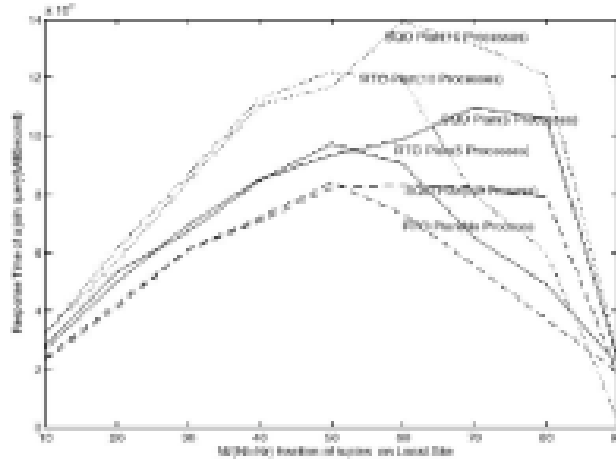


Figure 3(a): 10 Processes Running on Local Site*Figure 3(b): 15 Processes Running on Local Site*

For this experiment, RTO outperformed SQO by an average of 32.5%. Meanwhile, RTO incurred an average of 6.4% extra delays as compared to the optimal plan due to overhead of probe queries. We further reduced this marginal overhead of RTO, by employing our enhanced RTO. As expected, when the optional plan is simple join, the overhead cannot be avoided and both of RTOs behaved almost identical. However, an average reduction of 45% in overhead was observed for the cases where semi-join was the optimal plan.

In the second set of experiments, we spawned some processes (performing I/O's in cycles) on the local servers. Figures 3(a) and 3(b) demonstrate the performance of different optimizers when 10 and 15 processes are active on the local site, respectively. Recall that simple join performs one heavy join operation at the local site. Therefore, as the local site becomes more loaded, simple join becomes a less attractive plan. This behavior is illustrated in Figures 3(a) and 3(b) where the switching point (the point that simple join starts to outperform semi-join)

Figure 4: *Adaption of RTO to Workload Changes*

(this is due to the impact of $Delay_{send}(X)$ and $Delay_{receive}(X)$ factors in Eq. 10). Therefore, the simple-join plan will suffer as well. The beauty of our technique is that RTO does not need to take all these arguments into consideration in order to decide which plan to choose. The probe queries by measuring C_{l2r} and C_{r2l} , automatically capture all these behaviors. Therefore, as depicted in Figure 5, RTO can still choose the optimal plan.

CONCLUSIONS AND FUTURE DIRECTIONS

By implementing a sample distributed database system consisting of heterogeneous servers running homogeneous DBMS and connecting them via the Internet, the importance and effectiveness of run-time optimizations have been demonstrated. Our run-time *join* optimizer (RTO) issues two probe queries striving to estimate the cost of semi-join and simple join plans. By measuring the performance of the probe queries and analyzing the results, RTO selects an optimal plan taking into account run-time behavior of the environment at the time of query execution. We demonstrated through analysis and experiments that our RTO can capture the communication delay, server workload, and other hidden costs specific to certain implementation (i.e., RMI cost in our case). This is achieved without making any assumptions or attempts to model the chaotic behavior of the Internet-based environment. As a byproduct, our RTO is less sensitive to statistical anomalies than SQO. Furthermore, RTO relies less on the remote relation profiles than SQO since most of these information are captured during the probing process as a byproduct. Therefore, it becomes a better candidate for query optimization in multidatabase systems where the profiles resident on one site is not readily accessible to other sites. Finally, we proposed an *adaptive optimization technique* with RTO that captures sudden changes of run-time environment during the execution of query.

We intend to extend this work in four directions. First, we want to extend our experimental setup to multiple sites to extend RTO to support k -way joins. Second, we want to populate our object-relational database with multimedia data types in order to compare CPU intensive plans with communication intensive ones. We expect that here network congestion would have a high impact on preferring one plan to the other. Third, we would like to implement our adaptive optimization technique in order to capture sudden changes in run-time behavior. Finally, we want to run other DBMS softwares (e.g., DB2 and Oracle 8) on some of the sites in order to study our optimizer in a multidatabase environment.

REFERENCES

- Amsaleg, L., Bonnet, P., Franklin, M., Tomasic, A., & Urhan, T. (1997). Improving Responsiveness for Wide-area Data Access. *Data Engineering*, 20(3).

- Antoshenkov, G. (1993). Dynamic Query Optimization in rdb/vms. *Proc. of 9th IEEE Intl. Conference on Data Engineering ICDE*.
- Apers, P., Hevner, A., & Ya S. B. (1983). Algorithms for Distributed Query. *IEEE Transactions on Software Engineering*, 9(3).
- Bernstein, P., Goodman, N. & Wong E., Reeve, C., and Rothnie, J. (1981). Query Processing in a System for Distributed Databases. *ACM Transactions on Database Systems (TODS)*, 6(4) 602-625.
- Bodorik, P., Pyra, J., & Riordo, J. (1990). Correcting Execution of Distributed Queries. *Proc. of the Second Intl. Symp. On Database in Paralle and Distributed System*.
- Bodorik, P., Riordo, J. & Jacob, C. (1989). Dynamic Distributed Query Processing Techniques. *Proc. of ACM CSC'89 Conference*.
- Bodorik, P., Riordo, J. & Pyra, J. (1992). Deciding to Correct Distributed Query Processing. *IEEE Transactions on Knowledge and Data Engineering*, 4(3).
- Carey, M., DeWitt, D., Naughton, J., Asgarian, M., Brown, P., Gehrke, J., & Shah, D. (1997). The Bucky Object-Relational Benchmark. *Proc. Of ACM SIGMOD*.
- Ceri, S., & Pelagatti, G. (1984). *Distributed Databases Principles and Systems*, 141-156. McGraw-Hill.
- Chen, A., & Victor, O. K. Li. Improvement Algorithms for Semijoin Query Processing Programs in Distributed Database Systems. *ACM Transactions on Computers*, 33(11), 959 - 967.
- Chen, M. & P. Yu (1992). Interleaving a Join Sequence with Semi-joins in Distributed Query Processing. *IEEE Transactions of Parallel and Distributed Systems*, 3(5) 611-621.
- Cole, R. & Graefe, G. (1994). Optimization of Dynamic Query Evaluation Plans. *Proc. of ACM SIGMOD*.
- Evrendile, C., Dogac, A., Nural, S., & Ozcan, F. (1997). *Multidatabase Query Optimization. Journal of Distributed and Parallel Databases*, 5(1).
- Farley, J (1998). *Java Distributed Computing*, 189 - 225. O'REILLY.
- Haas, P.J. & Swami, A. (1995). Sampling-based selectivity estimation for joins using augmented frequent value statistics. *Proc. of 11th IEEE Intl. Conference on Data Engineering ICDE*.
- Informix. *Informix Universal Server: Informix guide to SQL: Syntax* (1997). volume 1 & 2 version 9.1,
- Kambayaashi, Y., Yoshikawa, M., & Yajima, S. (1983). Query Processing for Distributed Databases using Generalized Semi-join. *Proc. of ACM SIGMOD International Conference on Management of Data*, San Jose, CA.
- Mackert, L. & Lohman, G. (1986). R* optimizer validation and performance evaluation for distributed queries. *Proc. of 12th Intl. Conference on Very Large DataBases VLDB*.

- Olken, F. (1993). *Random Sampling from Databases*. Ph.D. thesis, University of California, Berkeley.
- Ozcan, F., Nural, S., Koskal, P., Evrendile, C., & Dogac, A. (1997). Dynamic Query Optimization in Multidatabases. *Data Engineering*, 20(3), 38-45.
- Paxson, V. (1997). *Measurements and analysis of End to End Internet Dynamics*. Ph.D. thesis, University of California, Berkeley, <http://www-nrg.ee.lbl.gov/nrg-papers.html>.
- Paxson, V. & Floyd, S. (1997). Why We don't Know How to Simulate the Internet. *Proc. of the 1997 Winter Simulation Conference*.
- Reese, G. (1997). *Database Programming with JDBC and JAVA*, 41-55, O'REILLY.
- Roussopoulos, N. & Kang, H. (1991). A Pipe n-way join algorithm based on the 2-way semi-join program. *IEEE Transactions on Knowledge and Data Engineering*, 3(4), 486-495.
- Shahabi, C., Khan, L., McLeod, D., & Shah, V. (1999). Run-time Optimization of Join Queries for Distributed Databases over the Internet. *Proc. of Communication Networks and Distributed Systems Modeling and Simulation (CNDS)* San Francisco, CA.
- Silberschatz, A., Korth, H., & Sudarshan, S. (1997). *Database System Concepts*, chapter 18, 587-631. McGraw-Hill.
- Urhan, T., Franklin, M., & Amsaleg, L. (1998). Cost-based Query Scrambling for Initial Delays. *Proc. of ACM SIGMOD*.
- Yu, C., & Meng, W. (1998). *Principles of Database Query Processing for Advanced Application*, chapter 3, 81-116.
- Zhu, Q. & Larson, P. (1994). A query sampling method for estimating local cost parameters in a multidatabase system. *Proc. of 10th IEEE Intl. Conference on Data Engineering ICDE*.

Chapter 7

Strategies for Managing EUC on the Web

R. Ryan Nelson
University of Virginia, USA

Peter Todd
University of Houston, USA

Beginning in the early 1980s, end-user computing (EUC) began to permeate organizations following the advent of the personal computer and a host of applications directed at the non-IS professional. Along with EUC came a whole new set of organizational opportunities and risks. Ten years later, the World Wide Web has opened the door to a yet more powerful set of EUC applications capable of reaching well beyond the boundaries of the organization. Indeed, Web technology permits end users to design applications that are immediately accessible by unlimited numbers of people from anywhere in the world. As a result, EUC using Web technology has introduced a whole new set of opportunities and risks for organizations. The purpose of this research is to examine what strategies organizations are using in their attempt to maximize the benefits of the Web for end users while mitigating the inherent risks. To this end, individuals from 12 major organizations were surveyed via the Web. The results indicate that while organizations seem to be doing an adequate job of establishing roles and standards, mechanisms for resource allocation, development management, and maintenance appear to be lacking. In fact, most firms seem to be relying on a monopolist control strategy at this point in time. While such a strategy may be the best approach

given the relative infancy of Web technology, it could prove to be an unstable strategy in the long run given the reach, range and flexibility of access that Web technology provides. Organizations are encouraged to take a proactive, formal posture toward EUC development on the Web.

“Build it and they will come” ... the marketing department of a Fortune 500 retailer spends several million dollars developing a “virtual store front.” They heavily promoted the initial system’s launch and eager consumers hit the system heavy and hard in its initial hours of operation. Unfortunately, the system infrastructure couldn’t handle the million plus hits they took in the first several hours. One customer sent in a comment to management that it appeared that, “this was a system running on a 286 in someone’s basement.” Not at all what was expected from this major retailer. Early problems with system access and use were so acute that usage dwindled over time. After six months, the system was getting only several hundred hits a day and generating only several thousand dollars in sales a month ... the system was shut down and the investment written off. Aside from the cost of the system, the firm noted that overall store sales had declined 4%, due in large part, to the damage to their reputation from the fiasco.

“Caution: user-developed Web sites can be hazardous to your organization” ... a senior manager within a financial services organization publishes a Web page with inaccurate stock quotes, financial information, and forecasts about various companies. This led to several hundred thousand dollars in sales to customers based on the erroneous information. When stock prices plunged, the firm was forced to make good on the losses to their customers. The manager lost his job.

“Breach of security” ... hacker uses a “sniffer” to steal 10,000 credit card numbers via a corporate Web site that was developed by an end user ... The system did not have security features in place to encrypt the transactions and they were sent as plain text. A large lawsuit is pending.

With the spread of end-user computing in the mid-1980s, organizations became concerned with how to manage the use of information technology by non-IS personnel. There was a need to both leverage and protect an organization’s information technology investments. This led to the development of various end-user computing (EUC) strategies which to varying degrees tried to balance the need for slack resources to foster end-user initiatives with the controls needed to protect organizations against risk (Davis, 1982). These strategies set standards and established policy over technology acquisition as well as the assignment of specific roles and responsibilities within organizations for EUC-related activities. They also

governed the planning, support and control tactics that were adopted within a given organization with respect to end-user computing.

Alavi, Nelson and Weiss (1987-88) identified five generic strategies that could be applied to the management of EUC in many organizations (see Table 1). These included the *laissez-faire* strategy, the monopolist strategy, the acceleration strategy, the marketing strategy and the operations-based strategy. The strategies differed in terms of their relative emphasis on expansion and control of EUC activities. The *laissez-faire* strategy, as the name

Table 1: Characteristics of Different Web Strategies (adapted from Alavi et al., 1987-88)

Strategies					
Characteristics	Laissez-faire	Monopolist	Acceleration	Marketing	Operations
Objective	“Do nothing”	Contain and restrict Web activities	Encourage and expand Web activities	Expand Web activities in certain form and directions	Obtain integration and efficiency in Web activities
Emphasis	“Hands-off” approach	Implementation of explicit controls	Provide support and broad-based education	Provision of value-added products and services	Standards
		Formal approval procedures	Highly responsive to end-user needs	Shaping demand	Formal cost/benefit analysis
Organizational structure	No formal structure	IS dept. active in Web containment and control	Centralized general support facility	Centralized facility for planning and coordinating	Centralized planning, prioritization, and monitoring
				Departmental support	Departmental support and enforcing standards and controls
Level of control	Very low	Very high	Relatively low	Relatively high	High

suggests, was a “do-nothing” approach that neither encouraged nor discouraged end-user activity while the monopolist and acceleration strategies focused on control and expansion, respectively. The more mature marketing and operations-based strategies emphasized controlled growth, targeted at specific objectives, pursued while adhering to established organizational standards. As organizations evolved in their approaches to EUC, they were expected to gravitate toward these more mature approaches.

In part, the evolution of strategies within an organization reflected the need to manage end-user resources differently over time. Typically, starting from either a laissez-faire or monopolist position, firms evolved through acceleration, marketing and operations-based strategies. In particular, the increasingly interconnected end-user environment meant that the technology platform had to be run in an operations-based mode that emphasizes resource planning, control and the implementation of standards. At the same time, encouraging the development of novel application content on top of the technology platform required innovative approaches that might best be fostered by marketing-oriented strategies. Thus, blended strategies that included different approaches to managing technology and content were required. In addition, controlling risk may have been less critical as technology became more stable and users, in general, became more educated about information technology issues. Furthermore, risks associated with user applications were generally bounded within a specific job task, functional area or business unit.

With the advent of Web-based technologies, end users have suddenly been given a more powerful set of tools to reach out beyond the organization and build system components that do not respect organizational boundaries. As illustrated by the three hypothetical scenarios described in the beginning of this article, the Web permits end-user applications to potentially have more profound affects on business processes, partners, and customers than ever before. Further, it implies that regulation of the basic technology infrastructure is no longer enough to ensure that end-user computing is well controlled. Basic hardware and software standards and the development of a technological infrastructure have put powerful tools in the hands of most end users that facilitate the development of applications and the dissemination of information beyond organizational boundaries. Thus, Web-based applications may lead to fragmentation in the methods by which a firm presents itself to external stakeholders, potentially affecting an organization’s image and relationship with both customers and suppliers.

This article examines the strategies and tactics that organizations are utilizing to coordinate and control Web-based development throughout the

end-user community. More specifically, we examine the way in which organizations are managing the diffusion of Web-related resources and systems developed by end users. The central question is whether comprehensive control and coordination strategies are being put in place or whether more piecemeal approaches prevail. The results of this inquiry should also help understand whether firms learn from earlier approaches, such as EUC in the 1980s, permitting them to establish more mature management approaches when radically new technologies are introduced. To this end, we examine the strategies being used by 12 large organizations. Our investigation looks at the 18 EUC management activities identified by Alavi, Nelson and Weiss (1987-88). These activities are reviewed in the next section.

WEB-RELATED MANAGEMENT ACTIVITIES

Specific tactics used to control Web-based development are summarized in Table 2. This set of tactics has been updated from those used to

Table 2. Web-Related Activities

1.	Acquisitions approval framework - the setting of procedures and requirements for formal approvals of, as well as economic justification of, Web-related tools and resources for use by end users.
2.	Technical standards - the setting of standards for Web-related hardware, software and communications technology purchased by end users.
3.	Developmental standards - the setting of standards for end-user development of Web applications (e.g., page design, navigation aids, etc.).
4.	Data management - the establishment of policies on data accessibility, reliability, consistency, and security related to Web applications in the end-user community.
5.	Assignment of roles and responsibilities - the establishment of policies for reducing role ambiguities between end users and information systems personnel with respect to Web-related applications.
6.	Scope of Web-related activities - the development of clear distinctions between the applications that can be developed by end users and those that should be developed by IS professionals.
7.	Setting priorities - for Web-related applications in the end-user community; e.g., order of development or resource allocation.
8.	Planning for equipment, capacity, and manpower - to ensure that sufficient resources for Web-related activities exist in the end-user community.
9.	Coordination across organizational boundaries - for the management of Web-related activities that cross functional lines (e.g., departments or divisions).
10.	Systems integration - planning for and facilitation of the technological interdependence between end-user and IS-developed Web applications.
11.	Training and education - of end-user personnel related to the development, management and use of Web-related technologies.
12.	Data access - supporting the end-user community's ability to obtain data for use by Web applications.
13.	Consulting - providing ongoing support services to end users in the area of Web-related technology.
14.	Financial controls and chargeback systems - for allocation and "fine tuning" of financial resources; may involve allocation (chargeback) of Web-related costs to end-user groups.
15.	Development - the design and implementation of Web-related systems by end users.
16.	Documentation - of end-user developed Web applications.
17.	Operation and maintenance - ongoing operation and maintenance of end-user developed Web applications.
18.	Audit and review - systems of checks and balances to ensure that appropriate controls and standards are developed, implemented, and adhered to by end users.

generically define end-user coordination and control strategies (Alavi et al., 1987-88). We group these factors into three general areas: standard setting, resource allocation, and applications development.

Standard setting relates to the definition of particular roles and standards that help to ensure that Web development efforts are carried out in a coordinated fashion that facilitates long term integration of systems. This includes setting up technical standards for hardware, Web development software tools and communications technology. In a Web-based environment, such standardization is important for setting up a common Web presence across user-developed applications. It also assists with the sharing of application components between users. Further, such standards are important to ensure the integration of applications. This is especially important as organizations move beyond having a simple information presence on the Web to having an interactive Web-based application that must be integrated with a company's application systems and business processes.

Data management and design standards take into account procedures for access to and security of corporate data. These data and design standards will also set rules for information presentation, reliability and accessibility. Information access and security are key issues for the development of Internet applications. Attention to security issues is critically important for most firms and is viewed as critical to ensuring user acceptance of interactive Web-based applications. Furthermore, information presentation and reliability are essential to establishing the value of corporate Web sites. Consistent presentation across end user Web applications is important to the overall image being projected by a firm. Reliability of information is also critical to the impression the Web site conveys to ultimate users.

It is also important in the Web-based environment to determine the appropriate assignment of roles and responsibilities among end-user developers and information systems personnel. Technical aspects of Web development technologies are complex and require access to scarce expertise. Coordinating and controlling the use of these resources is important for organizations trying to make effective use of their scarce technological expertise. Standard setting must also take into account the need for coordination of activities across organizational boundaries and the need for integration of Web-based applications into other business systems.

Establishing Roles and Standards- Hardware, Software, Data, People and Systems

- Technical standards - the setting of standards for Web-related hardware, software and communications technology purchased by end users.

- Data management - the establishment of policies on data accessibility, reliability, consistency, and security related to Web applications in the end-user community.
- Data access - supporting the end-user community's ability to obtain data for use by Web applications.
- Design standards - the setting of standards for end-user development of Web applications (e.g., page design, navigation aids, etc.).
- Assignment of roles and responsibilities - the establishment of policies for reducing role ambiguities between end users and information systems personnel with respect to Web-related applications.

Established standards are needed to facilitate the integration and deployment of Web-based technology in an effective manner. Once such standards are established, mechanisms are needed to help set priorities for resource allocation, determine the overall level of resource needs, and decide how to go about approving applications for development. As applications are developed and implemented, it is also important to have some set of financial controls that can be used to assess how costs are allocated among system participants. It is also necessary to complete post implementation audits and reviews to determine the payback from an implementation. The ways in which organizations attempt to support and control this resource allocation process will be important to determining the overall value of the organization's Web-based applications. For example, a complex and time consuming approval procedure may discourage the development of innovative applications by putting too many barriers between ideas and the application. At the same time, ensuring that user departments support their own Web-based application development will focus attention on the business bottom line, contributing to the likelihood that applications are a financial success. Such an approach may lead to an inherent conservatism that reduces the likelihood of novel, big-win applications.

Resource Allocation- Planning, Support and Control

- Setting priorities - for Web-related applications in the end-user community; e.g., order of development or resource allocation.
- Planning for equipment, capacity, and manpower - to ensure that sufficient resources for Web-related activities exist in the end-user community.
- Acquisitions approval framework - the setting of procedures and requirements for formal approvals of, as well as economic justification of, Web-related tools and resources for use by end users.

- Financial controls and charge-back systems - for allocation and “fine tuning” of financial resources; may involve allocation (charge-back) of Web-related costs across end-user groups based on resources employed.
- Audit and review - systems of checks and balances to ensure that appropriate controls and standards are developed, implemented, and adhered to by end users.

The process of applications development also needs to be managed in the context of a variety of factors that, taken together, will influence the way in which applications are built and the impacts, both positive and negative of those applications on the organization. First, it is important to determine how to establish the right mix of expertise on a project team and to delineate the types of applications that can be developed strictly within the user domain and those that require significant input from the IS group. The involvement of IS personnel will be most important for applications that cross organizational boundaries and require integration with existing operational and management systems. While it is tempting to craft an on-line ordering front end for a Web-based selling application, such an effort is of limited value, and potentially detrimental if it is not integrated with inventory, shipping, billing and production systems. Thus, a simple Web presence may be facilitated within a business unit, for that business unit, but the true benefits of Web-based applications require integration across a variety of systems.

In addition to looking at systems with an eye to overall integration and coordination, it is important for firms to consider the degree to which organizations have put programs in place for training and education with respect to Web-based technologies and the type of support the organization provides to assist users in the development of Web-based applications. These issues were critical to foster end-user computing in organizations, they are even more important in the context of Web-based technologies that allow users to easily reach beyond the boundaries of the organization and influence the views of external stakeholders.

Applications Development- Planning, Support and Control

- Scope of Web-related activities - the development of clear distinctions between the applications that can be developed by end users and those that should be developed by IS professionals.
- Development - the design and implementation of Web-related systems by end users.
- Coordination across organizational boundaries - for the management of Web-related activities that cross functional lines (e.g., departments or divisions).

- Systems integration - planning for and facilitation of the technological interdependence between end-user and IS-developed Web applications.
- Training and education - of end-user personnel related to the development, management and use of Web-related technologies.
- Consulting - providing ongoing support services to end users in the area of Web-related technology.
- Documentation - of end-user developed Web applications.
- Operation and maintenance - ongoing operation and maintenance of end-user developed Web applications.

WEB STRATEGIES

The Web-related activities outlined above can be combined to achieve, intentionally or unintentionally, a variety of Web development strategies. These strategies are summarized in Table 1. The laissez-faire strategy, which may be adopted implicitly, rather than explicitly, is a hands-off approach that allows for the evolution of various approaches to Web-based development across the organization. The monopolist approach takes the opposite viewpoint and attempts to strictly control Web-based activities in the organization and tries to ensure that a centrist viewpoints dominates development. Such a strategy will have complex formal approval procedures and controls that will limit individual action within the business units. The aim is to limit and control the evolution of Web-based applications and to ensure that a single well-coordinated Web presence is established. Such an approach is inherently low risk, but also is most likely to lead to mundane applications and limited innovation. The business risk that underlies such an approach is that monopolist organizations will fail to innovate sufficiently and will find themselves behind their competitors. Acceleration strategies are designed to encourage innovation and experimentation by limiting standards, making resources freely available, simplifying approval processes and providing high levels of user support. A marketing-based strategy is more mature. It focuses on the expansion of Web activities based on a well-developed and coordinated plan. Instead of making resources freely available, they are targeted at specific activities tied to business objectives based on a central planning model. Steering committees, chargeback schemes and other control mechanisms are likely in place to channel individual activity. Finally, the operations strategy suggests a factory like approach to development. Strict standards are in place, applications are evaluated based on strict ROI metrics, planning and monitoring of development activities will be centralized and enforcement of standards will be relatively strict.

The strategies described above suggest different ways that organization might manage Web development by end users. In the next section we examine data from 12 organizations to see how Web-based applications are being managed and controlled in practice within large U.S. organizations.

RESEARCH METHOD

Taking each of the activities described in section 2 we developed a Web-based survey to assess the degree to which organizations are employing each tactic (<http://www.commerce.virginia.edu/rrn2n/survey.htm>). The survey asked about both the extent to which the activity is performed and the respondent's view of the importance of that activity in the effective management of user developed Web-based applications. The survey consisted of 18 questions, each evaluated using a five-point Likert scales representing 1—not performed at all to 5—performed extensively and 1—extremely unimportant to 5—extremely important.

The survey was administered over the Web to a target sample of organizations with membership in two different university IS research centers. A single contact was made with a senior IS executive at each organization and they were asked to identify one or more members of their organization who would be most appropriate to respond to the questionnaire. Of the 20 organizations contacted, 12 responded to the survey request, for a response rate of 60%. The number of respondents per organization ranged from 1 to 11 (the latter being a large organization with multiple business units). The majority of respondents (28 of 34) were from the IS staff in their organization, with job titles such as IS manager, application development manager, Internet manager, systems engineer, technology manager, telecommunications manager and the like. Eleven respondents were middle managers and 15 were technical/professional staff. The remainder were in supervisory (5) or top management (2) positions. The respondents had, on average nine years of experience in the particular organization, with a range of 1 to 21 years. Of the 34 respondents, 16 had bachelor's degrees and 14 had graduate degrees.

RESULTS

Our Web development tactics are divided into three major categories: standard setting, resource allocation and development management. The results for each category are shown in both tabular (Tables 3, 4, and 5) and graphical format (Figures 1, 2, and 3). Each table/figure shows the degree to which that activity is performed within the organization as well as the

Table 3: Establishing Roles and Standards - Hardware, Software, Data, People and Systems

	Performance	Importance
Category Average	3.24	3.84
Technical standards	3.48	3.76
Data management	3.34	4.25
Data access	2.94	3.85
Design standards	3.27	3.73
Assignment of roles and responsibilities	3.15	3.61

Table 4: Resource Allocation

	Performance	Importance
Category Average	2.80	3.54
Setting priorities	3.09	3.88
Capacity planning	3.03	3.97
Approval framework	3.36	3.56
Financial control	2.33	2.82
Audits and reviews	2.18	3.47

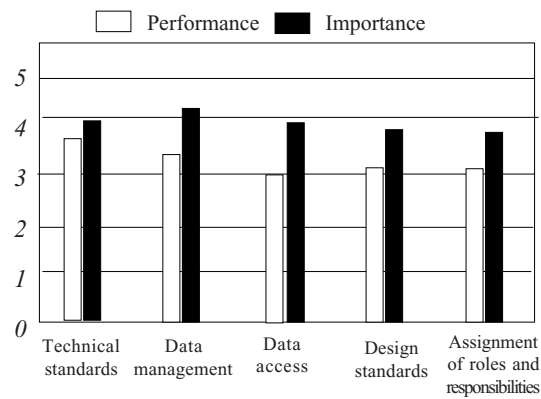
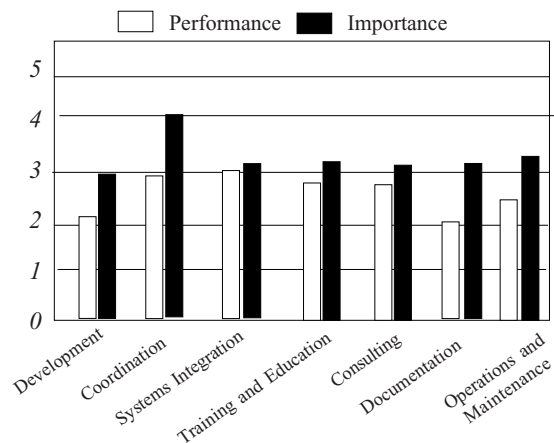
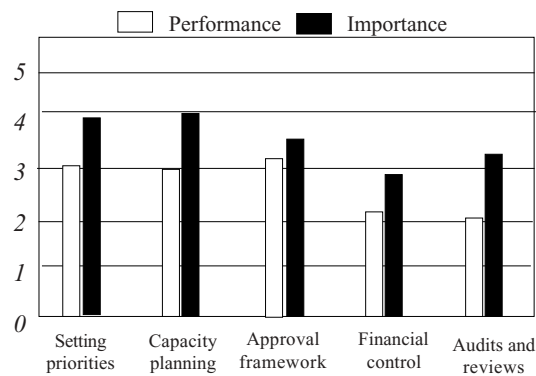
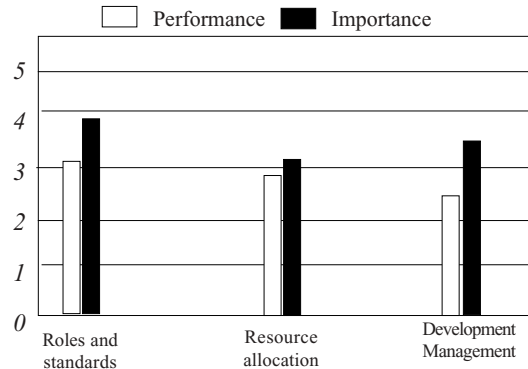
Table 5: Development management and support

	Performance	Importance
Category Average	2.64	3.46
Development	2.36	3.03
Coordination	2.85	4.00
Systems integration	2.88	3.36
Training and education	2.61	3.48
Consulting	2.70	3.58
Documentation	2.18	3.18
Operations and Maintenance	2.58	3.36

assessment of the relative importance of the activity to successful Web development.

Establishing Roles and Standards

With respect to standard setting, the mean response for all dimensions in the category is 3.24, suggesting that issues of standards have been dealt with to some extent with respect to Web-based development, but standards are not uniformly applied in all cases. At the same time, our respondents view standards as the most important element of a successful Web development strategy. More specifically, standards for data access seem to be the least frequently utilized (2.94) while technical standards seem to be most fully developed (3.48). The most important area of concern for the respondents was

Figure 1: Roles and Standards*Figure 2: Resource Allocation**Figure 3: Development Management**Figure 4: Category Comparison*

the need for standards with respect to data management (4.25), likely representing their concerns for security and information integrity in the Web-based world.

An analysis of response frequencies for each activity within this category seems to emphasize the fact that firms have reasonable control over technical standards and developmental standards, at least closely matching their perceived level of importance. They also clearly illustrate the perceived shortcomings in the areas of data management. While the majority of respondents rate the standard-setting areas at 4 or above on the importance scale most firms do not have mechanisms in place to provide this type of control. The same holds true for the formal assignments of roles and responsibilities between IS and end-user personnel. It would appear that increased attention to both of these areas is required.

Resource Allocation

In terms of resource allocation, our sample suggests that this set of management activities is performed less frequently with respect to the

development of Web applications, with an overall category average of 2.8 for the occurrence of these tactics and an average importance rating of 3.54. Within the categories there appear to be two distinct sets of responses. Relatively more use is made of priority setting (3.09), capacity planning (3.03) and approval frameworks (3.36) and relatively less use is made of financial controls (2.33) and audits (2.18). Interestingly, financial controls such as chargeback schemes are not frequently used, but are also not viewed as particularly important in this context. By contrast, despite the limited use of audits and reviews for applications, they are considered to be relatively important (3.47) by our sample of respondents. Perhaps the respondents were thinking of scenarios such as the ones described in the beginning of this article.

An analysis of response frequencies for each activity shows distinct gaps between the relative importance and use of priority setting and capacity planning mechanisms. It also highlights the serious discrepancies in the perceived importance versus application of audits and reviews of the development process. Together these results, consistent with those for setting standards, suggest a lack of systematic management of the Web-based development efforts across the organization.

Development Management and Support

Management of the development process is rated with an overall average of 2.64 in terms of the degree to which the specific functions are performed in the organization and are rated at 3.46 in terms of overall performance. In terms of the sub-categories, distinctions among different development roles (2.36), coordination across organizational boundaries (2.85), and planning for systems integration (2.88) appear to be carried out in only a limited way. Similarly training and education (2.61), support of user development (2.70), documentation of applications (2.18) and operations and maintenance issues (2.58) appear to receive limited attention. In each case these activities are viewed as relatively important. Noteworthy here is the assessment of the importance of coordination across organizational boundaries (4.0), relative to the degree to which it is carried out (2.85). Thus, it appears that although there is a recognized need for such coordination, there are only limited mechanisms in place to achieve it.

Analysis of response frequencies within this category highlights systematic shortcomings in terms of setting project scope, integrating applications and in particular, coordinating activities across organizational boundaries. They also show that there is a lack of systematic attention to training and support efforts to facilitate Web-based development. Further, the responses

for documentation and operations and maintenance suggest that there is only limited attention paid to managing the ongoing evolution of these applications within the organization.

Figure 4 provides a graphical comparison of the three categories of management activities. As previously discussed, the organizations surveyed seem to be placing the most emphasis on standards, followed by resource allocation, and then support.

OVERVIEW OF STRATEGIES

Table 6 provides an overview of the management activities and summarizes overall strategy by firm. Quick examination of the table shows that most firms seem to be following a “monopolist” Web management strategy. This strategic orientation manifests itself, in particular, with respect to the application of standards for hardware, development, data, and organizational roles and responsibilities. These control-oriented firms also tend to emphasize centralized resource planning and to a lesser extent the use of chargeback,

Table 6: Management Activities and Overall Strategy by Organization

Organization	Resource Allocation	Roles & Standards	Development Management	Strategy
Federal Agency	L	H	L	Monopolist
Non-Profit	H	H	H	Marketing
Telecommunications	L	H	L	Monopolist
Waste Management	L	H	L	Monopolist
Financial Services	L	L	L	Laissez-faire
Retail	L	H	L	Monopolist
Energy	L	L	L	Laissez-faire
Transportation	H	H	L	Monopolist
Energy	L	H	L	Monopolist
Telecommunications	H	H	L	Monopolist
Energy	L	L	L	Laissez-faire
Higher Education	L	L	H	Acceleration

audits and other evaluation techniques. These firms tend to provide limited support for end-user development, typically represented by training and consulting activities.

Three firms appear to utilize what is essentially, a *laissez-faire* strategy. These firms enforce little in the way of formal standards and only apply resource-planning mechanisms in a relatively limited fashion. At the same time they have little proactive support for user-based development, through training, consulting or related activities. Thus, they are not working actively to either encourage or discourage Web development by end users. Rather they appear to be taking more of a 'wait and see' attitude.

Finally, only two of the firms surveyed seem to be providing any significant support for end-user development of Web applications. These two firms differed from one another by how much control they exerted over the end-user community. While one organization actively encouraged user experimentation with Web-based applications (acceleration strategy), the other has tried to promote user development with specific scope and direction (marketing strategy).

DISCUSSION

The results of this study suggest that most organizations have chosen to utilize a monopolist strategy when it comes to Web-based end-user computing. More specifically, most organizations are attempting to contain and restrict Web development within the end-user community by implementing explicit controls and formal approval procedures while providing little in the way of direct support. Such a strategy suggests that organizations understand the risks inherent to Web applications and are attempting to mitigate those risks through centralized direction, development, and formal control mechanisms. In addition, the control orientation of these firms may simply reflect the fact that Web-based systems require an underlying network infrastructure which is relatively elaborate and will tend to be based on organizational standards.

The parallel between managing EUC in the late 1990s (i.e., via the Web) and managing EUC in the early to mid-1980s (i.e., via decentralized PC-based computing) is significant, but not surprising given the tendency to control widespread use of unproven technologies. This seems to be particularly true in a Web environment where end-user applications can have more profound affects on business processes, partners, and customers than ever before.

However, as noted by Alavi et al. (1987-88), the monopolist strategy may be unstable if adapted too early in the evolution of EUC and seems to break

down in companies that adopt it in an effort to curtail EUC activities altogether. A number of reasons may be given, including the following:

1. An increasingly Web-literate end-user population is demanding the capabilities and resources to directly develop some of its own Web applications (Munro et al., 1997).
2. With the constant improvement in Web application development environments, many end users are acquiring these tools out of local budgets in defiance of the corporate strategy (Kendall, 1997; Shah and Lawrence, 1996).

Based on the premise that EUC technology adoption follows a learning curve phenomenon, organizations are expected to evolve through the stages of acceleration, marketing, and operations. Consistent with such a pattern and the relative infancy of Web technology, none of the twelve organizations were deemed to have reached the operations stage.

Indeed, as new products appear, as the skills of end users increase, and as the competitive environment shifts, the priorities a company assigns to its various EUC Web applications appropriately evolve as well. As stated by one of the survey respondents:

“Employing intranet technologies will enable us to expand our application development community, allowing end users to play a more active role in shaping and constructing the tools they will use on a daily basis. This will have benefits in two primary ways: (1) systems will be cheaper, and (2) acceptance will be better, since the users have a higher stake in the success of the project..” - IS Manager, Energy Company

On the other hand, given the relatively high risk associated with Web applications, organizations would be wise to take a proactive, formal posture toward EUC development on the Web.

REFERENCES

- Alavi, M., Nelson, R., & Weiss, I.R. (Winter 1987-88). Strategies for end-user computing: an integrative framework. *Journal of Management Information Systems*, 4(3), 28-49.
- Davis, G. B., (1982). Caution: user-developed decision support systems can be dangerous to your organization. Presentation at the Fifteenth Hawaii International Conference on System Sciences, Honolulu, Hawaii.
- Kendall, K. (1997). The significance of information systems research on emerging technologies: seven information technologies that promise to improve managerial effectiveness. *Decision Sciences*, 28(4), 775-792.

- Munro, M. C., Huff, S. L., Marcolin, B. L., & Compeau, D. R. (1997). Understanding and measuring user competence. *Information & Management*, 33(1), 45-57.
- Shah, H. U. & Lawrence, D. R. (1996). A study of end user computing and the provision of tool support to advance end user empowerment. *Journal of End User Computing*, 8(1), 13-22.

Chapter 8

Exploring the Measurement of End User Computing Success

Conrad Shayo

California State University of San Bernardino

Ruth Guthrie

California Polytechnic University of Pomona

Magid Igbaria

Claremont Graduate University

As end user computing (EUC) becomes more pervasive in organizations, a need arises to measure and understand the factors that make EUC successful. EUC success is viewed as a subclass of organizational information system (IS) success, having distinct characteristics that distinguish it from other sources of organizational computing success. Namely, the success of applications developed by the information systems department (ISD), software vendors, or outsourcing companies. The literature shows that despite the volitional nature of end user computing, end user satisfaction is the most popular measure EUC success. Moreover, despite known limitations reported in the literature, self-reported scales are the instruments of choice by most researchers. This paper explores the literature on EUC success measurement and discusses the main issues and concerns researchers face. While alluding to the difficulty of devising economic and quantitative measures of EUC success, recommendations are made including the use of unobtrusive measures of success, take into account contextual factors, use well-defined concepts and measures and seek a comprehensive integrated model that incorporates a global view.

End user computing, defined as the optional development of computer applications and models by personnel outside the MIS department (Brancheau and Brown, 1991), is an important issue for IS executives (Niederman et. al., 1991; Watson and Brancheau, 1991). The emergence of EUC can be traced to the proliferation of microcomputers, increased organizational computing needs, more sophisticated user application development tools and higher computer and information literacy among staff and professional workers. Actual and invisible backlogs that could not be satisfied by the information systems department served as a catalyst to this trend. But, has IT investment in EUC been successful? Has the proliferation of microcomputers in organizations truly enhanced productivity, effectiveness and competitive advantage?

The answer to these questions should be seen in the context of overall computing success within the organization. A model showing subsets of organizational computing success and characteristics of application development within divisions or organizational computing is shown in Figure 1. The figure shows that overall organizational IS success is a conglomerate of end user developed applications (EUC success), information systems department (ISD) developed applications (ISD success), vendor off-the-shelf applications (vendor success), and applications developed by outsource companies (outsource success).

End user computing applications are usually developed with a great deal of freedom, using less standardization and control than ISD and vendor supplied applications. They often solve individual or departmental problems and are low risk but lack integration with other organizational systems. An organization's IS application's portfolio will be characterized by one or many intensities of each source of application development depending on the organization's IS acquisition strategy. The role of general management is to optimize the success of the application development mix by attempting to maximize the success of each component within the constraints of the organizational environment.

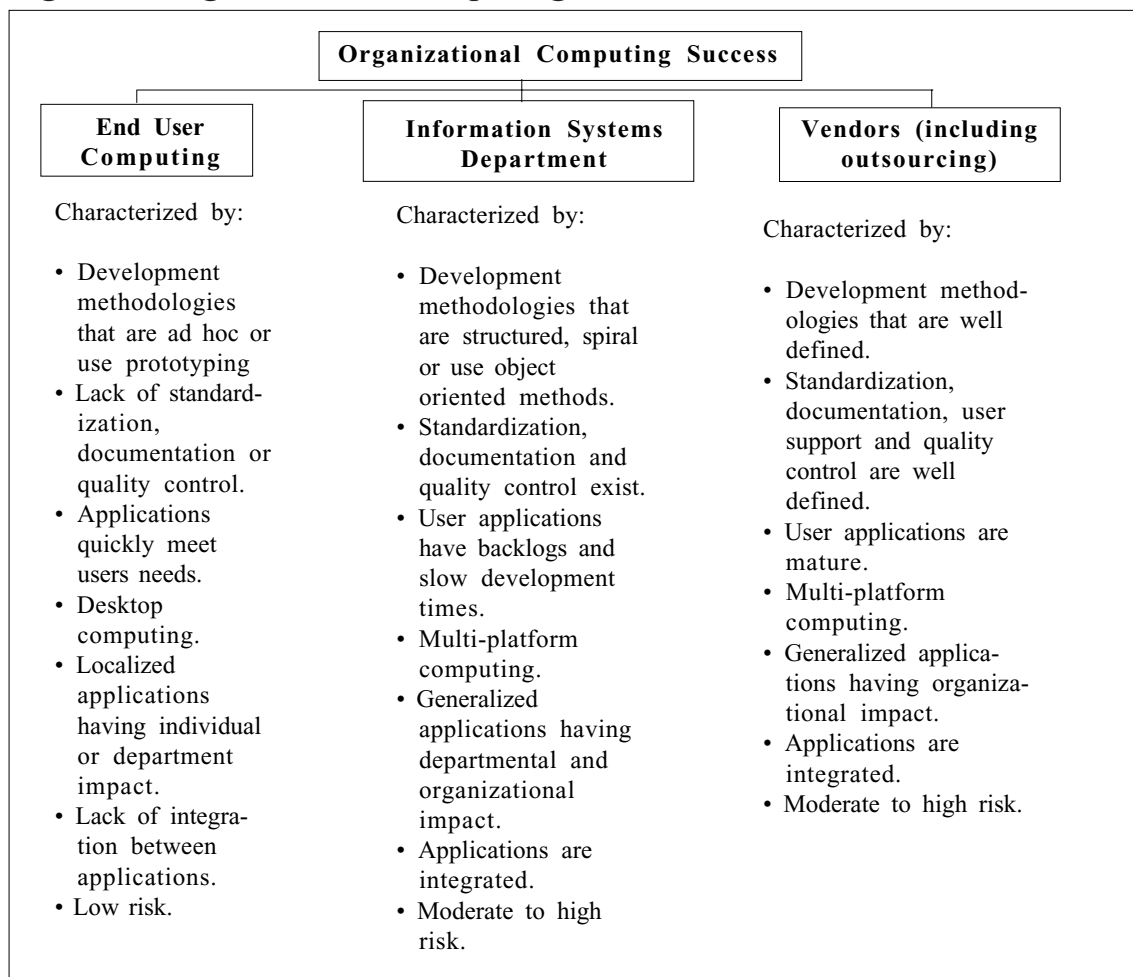
Measurement

Centuries ago, sailors would measure their speed and progress on the sea without the aide of a global positioning system. With a rope of evenly tied knots, the slow release of the rope into the water would give a measure of speed. It was a satisfactory measure of their progress toward their goal at the time. A captain, assuming he knew how to navigate, could judge progress by simply calculating the distance traveled. In the very early days of computing

and computer programming, measures such as lines of code, number of cards punched or graveyard hours at the lab were indicators of progress. Quantifying effectiveness—doing the right thing, and efficiency—doing something right was, and still remains, a more complex task.

This is true to the extent that the payoff from IT investment is continually under investigation (Panko, 1991; Markus, 1992; Brynjolfsson, 1993). Between the 1950s and early 1970s when mainframe computers were dominant, measurement of payoff from IT investment focused on number of jobs eliminated, costs avoided, or cost reduced and CPU hours used. General management was more concerned about efficiency at this time. With the introduction of mini and microcomputers between the mid-1970s and early 1980s, measurement focused on individual and work group effectiveness. General management was concerned that the rapid proliferation and investment in microcomputers was not paying off (Applegate, McFarlan, McKinney, 1996). The emergence of client/server computing and the Internet in the mid-1980s and early to mid-1990s, caused organizations to realize that IT could enhance or support organizational effectiveness and competitiveness.

Figure 1: Organizational Computing Success



This paper presents the findings and critique of recent literature on the measurement of EUC success. We explore EUC measurement issues related to individual, group, and organizational efficiency and effectiveness as well as organizational competitiveness. Note that EUC is the optional development of computer applications and models by personnel outside the MIS department. As shown in Figure 1, EUC success is just one contribution to the overall organizational computing success. The context of EUC also include the other sources of IS applications which include systems developed by the ISD and vendors. For the purposes of this paper, we define EUC success as the degree to which EUC contributes to individual, group, and organizational effectiveness and competitiveness in an environment that includes ISD and vendor developed applications.

The rest of the paper is organized as follows: first is a background review of the literature on IS success measurement in general and EUC measurement in particular. Second is a discussion of the problems of measuring EUC success. Third are conclusions and recommendations on how to improve the measurement of EUC success.

BACKGROUND LITERATURE ON INFORMATION SYSTEMS SUCCESS AND EUC SUCCESS

As indicated in Figure 1, a distinction is made between IS success and EUC success measures. IS success is an organizational computing success measure whereas EUC success is a more specialized individual computing success measure.

Several articles discuss components of IS success (Zmud, 1979; Ives and Olson, 1984; DeLone and McLean, 1992; Seddon, 1997), though its economic and quantitative measurement is often elusive. Consensus on specific measures of IS success seem to center around organizational impacts, system use and user satisfaction. DeLone and McLean (1992) identified six main categories of IS success: system quality, information quality, system use, user satisfaction, individual impact, and organizational impact. Figure 2 provides the temporal and causal interdependencies among the six variables. DeLone and McLean conclude that the six categories of success measures clearly indicate that IS success is a multidimensional construct and that IS success should be measured as such. As shown in Figure 2, "organizational impact" is seen as the ultimate measure of IS success. DeLone and McLean also suggest that user satisfaction should always be used when "IS use" is mandatory. Seddon creates an extension of the DeLone and McLean model

Figure 2: DeLone and McLean's Model of IS Success

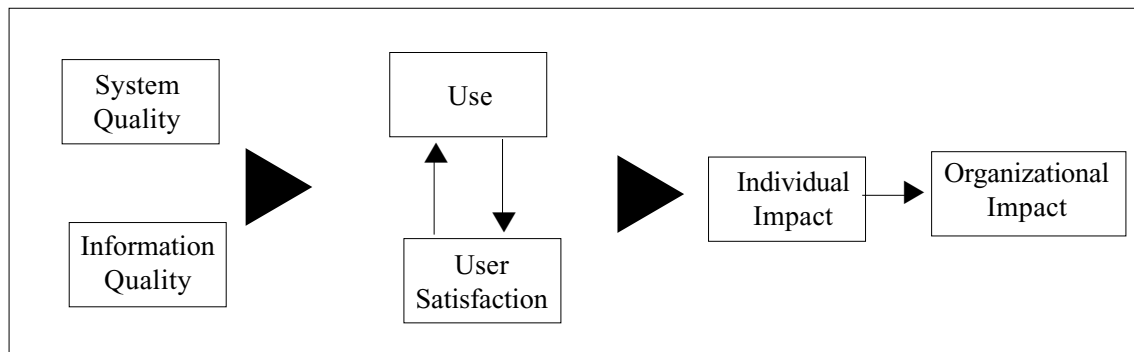
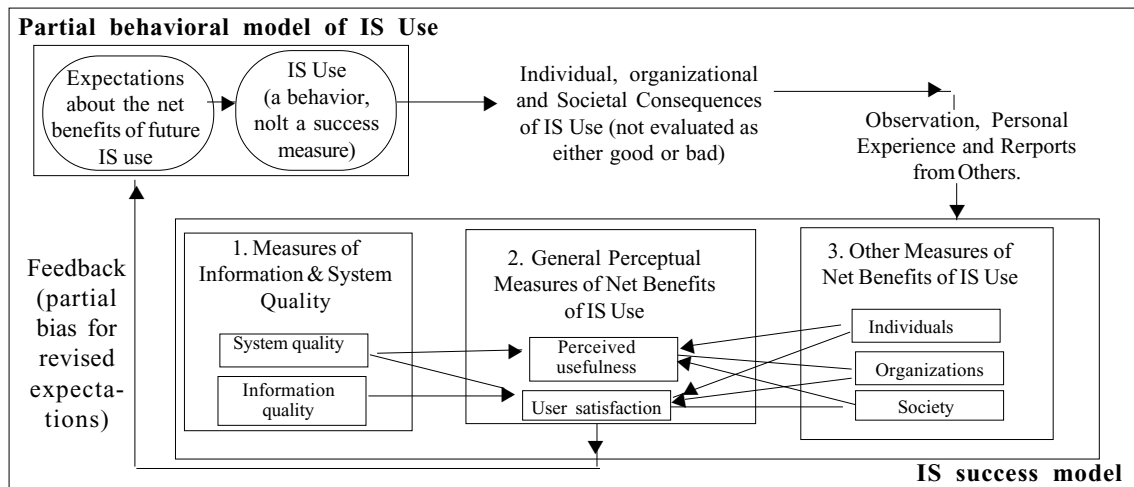


Figure 3: Seddon's Respecified Model of IS Success



in which behavioral aspects of IS use are separated from perceptual measures of IS success, Figure 3.

Seddon separates behavioral IS use from the DeLone and McLean model of IS success and divides it into expectations about net benefits of an introduced system and its actual use. He contends that actual use is a behavior, not necessarily a success measurement. The behavioral use creates individual, organizational or societal consequences that influence the IS success measures.

Seddon's motivation for expanding DeLone and McLean is to clarify variance and process measures in the model by elaborating the different meanings of 'use'. Use can refer to benefits that the system provides (perceived usefulness), anticipated benefits from future use (i.e. self efficacy, motivation), and use as part of organizational process (satisfaction, individual and organizational impacts). Moreover, Seddon argues that user satisfaction is paramount to measurement of information systems success. Application of Seddon's research model to EUC would suggest that the dependent variable should be "expectations about the net benefits of future use" of a specific end user developed application. End user computing satisfaction is used as the most important surrogate measure of EUC success.

Doll and Torkzadeh (1988) contend that general measures of end user satisfaction developed for traditional IS environments may no longer be appropriate for an end user environment where users directly interact with specific application software. The authors refer to such general traditional measures as “measures of user information satisfaction” (UIS). They propose that the term “end user satisfaction” be reserved for an end user’s satisfaction with a specific application. According to Doll and Torkzadeh, EUC satisfaction is defined as the attitude toward a specific computer application by someone who interacts with the application directly. They argue that the UIS instruments omit aspects important to EUC such as ease of use. In contrast, UIS measurement instruments, such as the Ives, et. al. (1983) instrument, measure general end user satisfaction with IS staff and services, information products, and user involvement rather than satisfaction with a specific application. Doll and Torkzadeh (1988) contend that, most end users can not evaluate such general UIS activities. They conclude that several IS staff and service items of the UIS instruments are less appropriate in an end user environment.

While research continues to grow on IS success, it remains scanty on the measurement of end user success. This is no surprise, considering that end user developed applications are rarely tracked formally by organizations. At the same time, it is not difficult to find organizations where an end user developed application is considered critical to daily operations. Furthermore, end users may be reluctant to allow measurement of the efficiency or effectiveness of their applications, especially from an outsider, for fear of job loss. Benign measures, such as end user satisfaction are less threatening and easier to obtain. However, this is problematic because users are asked to place a value on something about which they are far from objective.

According to Gerrity and Rockart (1986), EUC success will occur if there is:

- an increase of effectiveness in the individual using the developed application.
- a move to formalize the informal user system that was developed.
- an increase in learning on the part of the user in the ability to accomplish work.
- an increase in competitive advantage through support of new products, markets or opportunities.
- an improvement in organizational effectiveness as users access necessary data to improve decisions they make.

In 1990, Scott Morton added a sixth item to this list: EUC success is observed if an overall increase in national wealth due to increased knowledge of workers and information handlers exists.

MEASURING ELEMENTS OF EUC SUCCESS

At the advent of EUC in the late 1970s and early 1980s, metrics about end users kept by the IT department typically consisted of tallies of help desk requests sorted by hardware failures, packaged software assistance, laser printer maintenance and network connections for end user PCs. While these may be adequate measures of EUC operations and mean time between failure for hardware, they fall short in measuring end user computing success. The goals of end user computing are often hidden in a company and the speed and quality with which the goals are reached is hidden because end users often develop applications without organizational knowledge.

In our review of the literature on the measurement of EUC success we found that most articles could be categorized as having a focus on user satisfaction, use and productivity. The measurement instruments were largely based on subjective self reporting. The dependent and independent variable elements of EUC success varied depending on the objectives of the researchers. In the following section, we will use Seddon's extended DeLone and McLean's model to discuss the elements of EUC success. We focus on the IS Success Model components shown in Figure 3 which include: User Satisfaction, Perceived Usefulness, System Quality, Information Quality, Individual Impact, Organizational Impact, and Societal Impact.

User Satisfaction

User satisfaction is the most popular measure taken in recent studies. Instruments have been validated for both general (Bailey and Pearson, 1983; Ives, Olson and Baroudi, 1983) and specific (Doll and Torkzadeh, 1988) perceived measures of user information satisfaction. Other validated instruments end user satisfaction instruments include Doll & Xia (1996), Ives, Olson and Baroudi (1983), and Baroudi and Orlikowski, (1988). According to Amoroso and Cheney (1991), the Doll and Torkzadeh (1988) instrument is a more valid measure of EUC success than the Ives et al. (1983) instrument.

The Doll and Torkzadeh (1988) instrument for measuring EUC satisfaction requires subjective self-reports of content, format, accuracy, ease of use, and timeliness of an application. Studies using user satisfaction as a measure indicate that EUC support and policy are correlated with satisfaction (Berferon and Berube, 1988), and that users are more satisfied with microcomputers than mainframes (Glorfeld and Cronan, 1992). Evidence also links satisfaction to user skill (Glorfeld and Cronan, 1992; Harrison and Rainer, 1992, Barker, 1994), information quality (Doll and Torkzadeh, 1988) and motivation (Barker 1994; Igbaria, Parasuraman and Baroudi, 1996).

Several articles discuss the merits and problems with measurement of user satisfaction as an indicator of EUC success (Galletta and Lederer, 1989; Torkzadeh and Doll, 1991; Etezadi-Amoli and Farhoomand, 1991; DeLone and McLean, 1992, Doll et al., 1994). Seddon (1997) would argue that for lack of a better measure, user satisfaction is the most desirable measure of net benefits or success. However, this is problematic in that equating user satisfaction with EUC success does not tell us whether the system is productive or whether its use gives the concerned organization some economic gain. Consider a user-developed program that produces reports that are redundant to those produced by another organizational computing system. While the user may be highly satisfied, the overall impact could be described as a failure. There is, therefore, need to measure correlates of individual end user satisfaction with organizational context variables.

Bergeron and Berube (1988) correlated perceptions of IS support features with satisfaction. They found a negative correlation between user satisfaction and the number of micro-computing policies. Glorfeld and Cronan (1992) used both the Ives, et. al. and the Doll and Torkzadeh measures of UIS to study the success of EUC management techniques. Results indicated a positive relationship for the impact of management technique on satisfaction.

Information Systems Use

Measures of information system use are blurred in the literature because it is difficult to sort objective from subjective measures and to distinguish when use is mandatory or voluntary. Obviously, if the use is mandatory, satisfaction might be a better measure of IS success. Seddon (1997) tries to clarify the many types of IS use by distinguishing (a) expectations about the net benefits of future use of an application developed by an end user, i.e., perceived usefulness of future use, and (b) actual use of the individual application. See Figure 3.

Expectations of future net benefit is distinguished from perceived usefulness and user satisfaction in that it is a measure of an individual's expectation about the benefits of future use of the information system. For example, end users will be asked to evaluate the statement: "Using [a specific application] will improve my job performance", where 1 = strongly disagree, 3 = uncertain, 5 = strongly agree." This is related to an individual's goals, self-efficacy and level of experience or skill. In the literature, measures of expected benefit follow a model from Davis (1989) and indicate the relationship between perceived usefulness and actual use (Segars and Grover, 1993; Taylor and Todd, 1995).

Actual use of an information system is one of the most frequently reported measures of IS success (DeLone and McLean, 1992). Measures include: observing microcomputer monitors and self-reported actual use. According to Delone and McLean, usage, whether actual or perceived, is a useful measure of IS success only when such use is voluntary. Cheney, et. al. (1986) agree by proposing that unless use is mandatory, an end user will utilize EUC facilities only when they are perceived to be of value to the user. Thus, the authors recommend application utilization as a surrogate measure of EUC success when use is voluntary. The measurement of perceived usefulness in the Seddon's (1997) extended model refers to a user's reaction to a system that has already been introduced and used. End users are asked to evaluate the statement: "Using [a specific application] has improved my job performance", where 1 = strongly disagree, 3 = uncertain, 5 = strongly agree."

In a study of motivation on microcomputer usage, Igarria, Parasuraman and Baroudi (1996) found that perceived usefulness was the strongest motivator for system acceptance. Evidence also showed that skill played a major role in microcomputer acceptance. In a study of system effectiveness and use (Snitkin and King, 1996), usage and perceived usefulness were also highly related. Moreover, people with high analytic ability are more frequent users. Marcolin, Munro and Campbell (1997) found similar results with the addition of computer anxiety as a variable. Ease of use was also correlated with actual use (Adams et al., 1992). Ability was later shown to influence variety of tasks and system use (Guimaraes and Igarria, 1997) and to be directly related to efficacy, performance and job satisfaction (Henry and Martinko, 1997).

A few researchers have called for the need to concentrate on end user competence and the quality of the applications they develop. Munro, Huff, Marcolin and Compeau (1997) developed a measure for end user competence consisting of breadth and depth of knowledge and end user finesse. The authors concluded that there was need for a better measure of competence in order to determine if investment in end user technologies and training was warranted. Plavia's (1991) lab experiment found that command level users working with databases performed better when presented with data models showing their own view of reality. They found that no particular method for program development resulted in higher quality applications designed by end users. Pseudocode and direct writing proved to be most productive. Edberg and Bowman (1996) found that in a controlled experiment, students posing as surrogate IS professionals also produced higher quality applications and were more productive than end users. Both studies are evidence of Plavia's warnings that a massive amount of training may be required to make end users

productive in systems development. Both studies used individual self reporting and objective measures to quantify their findings.

Amoroso and Cheney (1992) suggest the need for researchers to focus on the quality of end user developed applications. The authors define quality as the degree to which an application attains its goals from the perspective of the user. They recommend that researchers should combine end user computing satisfaction and application utilization measures to assess the quality of user developed applications. The variables included in the Amoroso and Cheney instrument include: reliability, effectiveness, portability, economy, user friendliness, understandability, verifiability, and maintainability. This means that an end user-developed application that ranks higher on the above measures will in turn increase end user utilization of the application and satisfaction.

If we consider systems developed outside the traditional MIS department, evidence of measures of system quality is rare. Traditionally, IS quality has been measured by evaluating program reliability, user interface design, accuracy or use. In the end user computing literature, measures focus primarily on use of systems and user skills. Measures of EUC information quality are infrequently used. Presumably, if an end user is developing their own system, they have expert knowledge of the information, its timeliness and degree of accuracy that is necessary. Saleem (1996) gives support of this assumption in a series of controlled studies of user participation in IS design. The author found that user participation has a positive impact on development, if their domain knowledge is adequate. Saleem concludes that since user's expertise is invaluable to the design effort, user management need to examine which phases of development the user should be involved in.

Measures of Individual and Organizational Impact

According to DeLone and McLean (1992), individual impact is the most difficult category to define in unambiguous terms. For example, the individual impact of an end user developed application could be related to a number of different measures such as impact on performance, understanding, decision making and/or user activity. Blili (1992) developed an end user computing impact measure that assessed managerial performance, productivity and job satisfaction. Blili's impact measures were very similar to both UIS and IS actual use measures.

Measures for organizational and societal process impacts are rare. Brown and Bostrom (1994) found evidence that EUC management strategy can support different growth objectives. The authors conclude that MIS managers and researchers should pay more attention to the degree of organi-

zational centralization, formalization, and complexity when they evaluate the effectiveness of end user computing management in an organization. Hitt and Brynjolfsson (1997) showed that increased investment in IT is related to distributed management structures.

Other aspects of EUC success include support, skills, and task characteristics. Rainer and Harrison (1992) developed and validated the EUC activities scale that gives a mechanism for classifying specific computing work done by end users. Mirani and King (1992) developed an instrument to measure levels of EUC support, arguing that higher support levels will better promote EUC within organizations.

End user support and skills have been studied to determine their influence on system use and adoption of new technology. Bowman et al. (1993) found that colleagues and software manuals provided the majority of end user support. The author's findings were based on a sample of twelve organizations. The authors also found that computing skill, position and personal characteristics had no correlation with the type of support chosen by and end user. Mirani and King (1994) found that information centers do not assess user needs in attempting to provide support. The authors found that when support needs were provided, user satisfaction increased.

A complete list of the independent variables found in the articles identified in Table 1. Table 2 lists the dependent variable.

PROBLEMS WITH EUC SUCCESS MEASUREMENT

Measuring EUC success seems to be an intractable problem. Studies contribute to our understanding of EUC success, yet they lack consistency in measures, design and technology to gain larger understanding and insight. Problems associated with the measurement of EUC success are:

1. *Control and Clarity* - There is a need to control for task, technology and context in studies that measure EUC success. Considerations for task variety and complexity were rarely made in the literature. Given the wide variety of skill, position and types of computing work, it is imperative that we consider controlling for task in the measurement of EUC success. Similarly, wide ranges exist between the technology used in research. Surely, a fax machine has qualitatively different attributes than e-mail or virtual chat. Indeed, a menu driven e-mail package (used at many Universities) is nothing like AOL-mail (used in many homes). The problem of control is exacerbated by rapid changes in technology that make it difficult to repeat similar tests and measures over time. Context needs to be clearly defined so it is understood

Table 1: Measures used as the Independent Variable in EUC Success

Management Support for Planning and Control <ul style="list-style-type: none"> * Top management understanding of IT * Development of appropriate strategies/policy * Top management integration of the organizational microcomputer strategic plan with the IS master plan * Provide a budget for training programs in-house or at remote location by company trainers (software training, OS training, communications training) * Provide a budget for educational programs in-house or at remote location by company personnel (general computer literacy, functional computer literacy) * Encouraging experimentation with microcomputers * Encouraging use of IT to support a wider variety of business tasks * Rewarding efforts of using IT to meet set goals at sectional, department, divisional, and corporate levels * Developing a core of internal experts who will train others (local resident experts) 	User Characteristics <ul style="list-style-type: none"> * Years of Education * Cognitive Style * Command Level skills * Programming skills * Self-efficacy * Demographics: gender, age * Inputs consumed to provide outputs: programming time, flow diagram, pseudocode, narrative description, direct writing, 4GL * Personality * Computer attitudes * Computer anxiety * Math anxiety * Experience * Skill variety * Autonomy * End user computing sophistication/competence: ability, usage intensity, application customization * Self-efficacy
Other types of support <ul style="list-style-type: none"> * Training provide by other colleagues * Providing software library services * Access to a information center (IC), help desk or hotline * Maturity of help desk to support end users 	System Characteristics <ul style="list-style-type: none"> * High end * Low end * Quality: Security, Functionality, Ease of use, Documentation * Type of Application * Value and Usefulness of system terminology
IC Support for Hardware and Software <ul style="list-style-type: none"> * Guidance on selecting hardware * Guidance on selecting software * Hardware setup/configuration * Software installation * Backup/Recovery 	Information Characteristics <ul style="list-style-type: none"> * Quality of output: content, structure, correctness, accuracy, format, ease of use, timeliness * Number of defects/function point * Quality attribute models * Value and Usefulness of screen displays
IC Support for Application Development <ul style="list-style-type: none"> * Development assistance * Access to corporate data * Assist users with finding data * Application maintenance * Troubleshooting 	User/System/Task Interaction <ul style="list-style-type: none"> * End user participation in Analysis and Design * End user involvement in Analysis and Design: perceived risk, degree of pleasure, status value * End user usage of the system * Time on project in task categories * LOC/hour * Function points/hour * Outcome expectancy * Perceived usefulness * Perceived fun * Satisfaction
Organizational Structure <ul style="list-style-type: none"> * Centralized * Decentralized 	

Table 1: Measures used as the Independent Variable in EUC Success (continued)

External Support <ul style="list-style-type: none">* Good relationships with external hardware and software vendors or consultants breeds positive feelings, realistic expectations* Technical support* Training provided in a remote location or on company premises by external consultants, friends, vendors, or educational institutions* Educational programs provided in a remote location or on company premises by external consultants, friends, vendors, or educational institutions	Task Characteristics <ul style="list-style-type: none">* Task identity* Task significance* Task uncertainty: complexity, volatility Other characteristics <ul style="list-style-type: none">* EUC growth in the organization* Societal pressure
--	---

Table 2: Measures used as the Dependent Variable in EUC Success

End User Satisfaction
End User Productivity
End User Computer Skill
End-User Ability/Competence
End User Success
Motivation for System Use
Work Effectiveness
IS Acceptance
Job Performance
EUC Management Effectiveness
System Effectiveness
System Usage

whether use of the information system is mandatory or voluntary or whether a competing alternative information system exists. Considerations of importance, strategic value and organizational level must also be taken into account. Researchers must be diligent in clearly defining the context of the study and the use of the IS and in controlling for task complexity and type of technology used.

2. *Creation of Meta and Longitudinal Data* - There is a shortage of studies that have any data of a longitudinal nature. Most studies identified in the literature are cross sectional and not of pre-post nature. Task, technology

and context variety is keeping us from gaining longitudinal insight about individuals (measuring how individuals change as they develop higher technology skills) and insight about organizations (measuring the impacts of technology changes over longer periods of time). This is exacerbated by rapid changes in technology, and by the undocumented, hidden nature of end user developed applications. The lack of repeatability again hinders researchers in performing meta-analysis from several studies.

3. *Unit of Measurement*—Most studies in the literature tend to focus on the individual end user as the unit of analysis in EUC measurement. However, if we are concerned with overall organizational success, group, departmental and organizational measures of EUC success need to be applied. For instance, if satisfaction is a measure of EUC success, success should be measured at many levels. A CEO or departmental manager may have a very different perspective on an IS success than an end user has.

Moreover, the reason for the existence of EUC has changed. Actual and invisible backlogs in the 1980s were the prime reasons for the emergence of the EUC phenomenon. However, in the 1990s, EUC is seen as part of organizational computing strategy, requiring management evaluation. It is rare to find EUC measuring instruments at the divisional and organizational levels. One main reason is that EUC is mostly an individual activity and it is easier to obtain self-reports from the individuals. However, the portfolio approach suggested in this paper calls for organizational measures that will evaluate the efficiency of a given EUC strategy. There is, therefore, need to devise EUC measurement instruments that reflect this change. Since EUC is mostly individual, there is a lack of a general organizational view and measurement of the impact of EUC on overall organizational computing. There is need for measures that will allow managers and end users to set goals, allocate organizational computing resources effectively and eventually optimize the mix of computing in organizations. Also, the granularity of the measures posits a problem. For example, perceived usefulness could be evaluated at the future level or at the current level.

4. *Lack of objective measures of end user performance in a field setting.*
- End user computing activities are rarely visible to the rest of the organization. Such activities are therefore difficult to observe, document and measure unobtrusively.

5. *Need for a more comprehensive and integrated model.* Most studies focus only on a few antecedents while ignoring others. For example, the Doll and Torkzadeh (1988) instrument focuses mostly on information quality factors while ignoring system quality, perceived net benefits of actual use. There is need to develop a more comprehensive, integrated network model to

assist in adding clarity to the future study of EUC success. Currently, a lack of agreement exists on the direction and granularity of the causal variables of EUC success. A case in point is Seddon's extension of DeLone and McLean. Whereas IS use and user satisfaction are the antecedents to the benefits accruing to the individual (individual impact) and the organization (organizational impact) in the DeLone and McLean model, it is the reverse in the Seddon's extension. It is, therefore, important to realize that there will always be a feedback loop that changes the direction of the causal relationships. Researchers should therefore acknowledge that some measures may covary and none causes that other (Seddon, 1997).

6. *Lack of conceptual definitions*— The operational definition of the measuring variables must correspond with the conceptual definition. Researchers should develop and use measures that are well established and validated in IS and other disciplines. For example, are we really measuring end user satisfaction? How is this different from satisfaction with objects researched in other fields such as organizational behavior and marketing?

7. *Lack of a global view* — Most EUC research has been conducted in North America. There is belief that the results obtained in the North America setting are generalizable to other countries. This belief may be ill advised given differences in culture, socio-work roles, level of IT sophistication and access to technology. Models of EUC need to be checked for external validity across cultures. This may prove important in the future as more companies employ a global workforce.

CONCLUSION

The need for a comprehensive, integrated model of IS success is apparent. If such a model existed, concepts and measures could evolve into well-defined instruments for researchers to use. When well-defined concepts and measures are at hand, studies can converge on an overall understanding of EUC success and build upon related works to expand the body of knowledge. Once we build measures upon common concepts, we can begin to accumulate a body of knowledge that validates our measurement tools. Well-established and validated instruments can add great clarity to our understanding of all research. Measures that are direct and uncoupled from other multi-attribute constructs can lead to more generalizable results.

There is also a great need to pay attention to the contextual factors of end user computing. Too many studies are focused at the individual level, ignoring departmental, work-group, organizational or even global effects. A

broader view of the implications of EUC in organizations, beyond individuals, could give great insight into productivity, quality and competition.

REFERENCES

- Adams, D. A., Nelson, R. R. & Todd, P. A. (1992) Perceived usefulness, ease of use, and usage of information technology: a replication. *MIS Quarterly*, 16(2), 227-247.
- Amoroso, D.L. & Cheney, P.H. (1991). Testing a causal model of end-user application effectiveness. *Journal of Management Information Systems*, 8(1), 63-69.
- Amoroso, D.L. & Cheney, P.H. (1992). Quality end-user developed applications: some essential ingredients. *DATA BASE* 23(1), 1-11.
- Applegate, L.M., McFarlan, F.W. & McKinney, J.L. (1996). *Corporate Information Systems Management: Text and Cases*, Fourth Edition, Chicago: Irwin.
- Bailey, J.E. & Pearson, S. (1983). Development of a tool for measuring and analyzing user satisfaction. *Management Science*, 29(5), 530-545.
- Barker, R. M. (1994). The interaction between end user computing levels and job motivation and job satisfaction: an exploratory study. *Journal of End User Computing*, 7(3), 12-19.
- Baroudi, J.J., & Orlikowski, W. J. (1988) A short-form measure of user information satisfaction: a psychometric evaluation and notes on use. *Journal of MIS*, 4, 44-59.
- Bergeron, F. & Berube, C. (1988). The management of the end-user environment: an empirical investigation. *Information & Management*, 14, 107-113.
- Blili, S., Raymond, L. & Rivard, S. (1998). Impact of task uncertainty, end-user involvement, and competence on the success of end-user computing. *Information & Management*, 33, 137-153.
- Blili, Samir, Raymond, Louis, and Rivard, Suzanne (1994). Definition and measurement of end user computing sophistication. *Journal of End User Computing*, 8(2), 3-12.
- Bowman, B., Grupe, F. H., Lund, D. & W. D (1993). An examination of sources of support preferred by end-user computing personnel. *Journal of End User Computing*, 5(4), 4-11.
- Brancheau, J. C. & Brown, C. V. (1993). The management of end-user computing: status and directions. *ACM Computing Surveys*, 25(4), 437-482.

- Brown, C. V. & Bostrom, R. P. (1994). Organization designs for the management of end-user computing: re-examining the contingencies. *Journal of Management Information Systems*, 10(4), 183-211.
- Brynjolfsson, E. (1993). The productivity paradox of information technology. *Communications of the ACM*, 36(12), 67-77.
- Cheney, P.H., Mann, R.I. & Amoroso, D.L. (1986). Organizational factors affecting the success of end-user computing, *Journal of Management Information Systems*, 3(1), 65-80.
- Davis, F. D. (1989). Perceived usefulness, perceived ease of use and user acceptance of information technology. *MIS Quarterly*, 13(3), 319-340.
- DeLone, W. H. & McLean, E. R. (1992). Information systems success: the quest for the dependent variable, *Information Systems Research*, 3(1), 60-95.
- Doll, W. & Torkzadeh, G. (1988). The measurement of end user computing satisfaction. *MIS Quarterly*, 12(2), 259-274.
- Doll, W. J. & Torkzadeh, G. (1991). The Measurement of End-User Computing Satisfaction: The Theoretical and methodological issues. *MIS Quarterly* 15(1), 5-10.
- Doll, W. J. & Xia, W. (1997). A confirmatory factor analysis of the end-user computing satisfaction instrument: a replication. *Journal of End User Computing*, 9(2), 24-31.
- Doll, W. J., Xia, W. & Torkzadeh, G. (1994). A confirmatory factor analysis of the end-user computing satisfaction instrument. *MIS Quarterly* 18(4), 357-369.
- Edberg, D. T. & Bowman, B. J. (1996). User-developed applications: an empirical study of application quality and development productivity. *Journal of Management Information Systems*, 13(1), 167-185.
- Etezadi-Amoli, J. & Farhoomand, A. F. (1991). On end-user computing satisfaction. *MIS Quarterly* 15 (1), 1-4.
- Galletta, D.F. and Lederer, A.L. (1989). Some cautions on the measurement of user information satisfaction. *Decision Sciences*, Vol. 20, 419-438.
- Gerrity, T. P. & Rockart, J. F. (1986). End-user computing: are you a leader or a laggard. *Sloan Management Review*, 27(4), 25-34.
- Glorfeld, K. D. & Cronan, T. P. (1992). Computer information satisfaction: a longitudinal study of computing systems and EUC in a public organization. *Journal of End User Computing*, 5(1), 27-36.
- Guimares, T., Igbaria, M. and Lu, M. (1992). The determinants of DSS success: an integrated model. *Decision Sciences*, 23, 409-430.
- Harrison, A. W. & Rainer, R. K. (1992). The influence of individual differences on skill in end-user computing. *Journal of Management Information Systems*, 9(1), 93-111.

- Henry, John w. and Martinkio, Mark J. (1997). An attributional analysis of the rejection of information technology. *Journal of End User Computing*, 9(4), 3-17.
- Hitt, L. M. and Brynjolfsson, Erik (1997) Information technology and internal firm organization: an exploratory analysis. *Journal of Management Information Systems*, 14(2), 81-101.
- Igbaria, M. Parasuraman, S. and Baroudi, J.J. (1996) a motivoinal Model of Microcomputer usage. *Journal of Management Information Systems*, 13(1), 127-143.
- Ives, B. & Olson, M. (1984). User involvement and MIS success: a review of research. *Management Science*, 30(5), 586-603.
- Ives, B., Olson, M. H. & Baroudi, J.J. (1983) The measurement of user information satisfaction. *Communications of the ACM*, 26(10), 785-793.
- Marcolin, B. L. Munro, M. C., Campbell, K. G. (1997). End user ability: impact of job and individual differences. *Journal of End User Computing*, 9(3), 3-12.
- Markus, L., & Soh, C. (1992). Banking on Information Technology: Converting IT Spending into Firm Performance. Working Paper, University of California, Los Angeles.
- Mirani, Rajesh and King, William R. (1994). Impacts of end-user and information center characteristics on end-user computing support. *Journal of Management Information Systems*, 11(1), 141-166.
- Munro, M. C., Huff, S. L., Marcolin, B. L., Compeau, D. R. (1997). Understanding and measuring user competence. *Information & Management*, 33, 45-57.
- Niederman, F., Brancheau, J.C., and Wetherbe, J.C. (1991). Information Systems Management Issues for the 1990s. *MIS Quarterly*, 15(4), 474-500.
- Panko, R. R. (1991). Is office productivity stagnant? *MIS Quarterly*, 15(2), 191-203.
- Plavia, P. (1991). On end-user computing productivity. *Information & Management*, 21, 217-224.
- Rainer, R. K. & Harrison, A. W. (1992). Toward development of the end user computing construct in a university setting. *Decision Sciences*, 24(6), 1187-1202.
- Saleem, N. (1996). An empirical test of the contingency approach to user participation in information systems development. *Journal of Management Information Systems*, 13(1), 145-166.
- Scott Morton, M.S. (1991). Introduction. In Michael S. Scott Morton (ed.), *The Corporation of the 1990s: Information Technology and Organizational Transformation*, New York: Oxford University Press.

- Seddon, P. B. (1997). A respecification and extension of the DeLone and McLean model of IS success. *Information Systems Research*, 8(3), 240-253.
- Segars, A. H. & Grover, V. (1993). Re-examining perceived ease of use and usefulness: a confirmatory factor analysis. *MIS Quarterly*, 17(4), 517-525.
- Snitkin, S. R. & King, W. R. (1986). Determinants of the effectiveness of personal decision support systems. *Information & Management*, 10, 83-89.
- Taylor, S. & Todd, P. A. (1995). Assessing IT usage: the role of prior experience. *MIS Quarterly*, 19(4), 561-570.
- Torkzadeh, G. & Doll, W. J. (1991). Test-retest reliability of the end-user computing satisfaction instrument. *Decision Sciences*, 22, 26-37.
- Watson, R.T. & Brancheau, J.C. (1991). Key issues in information systems management: an international perspective. *Information & Management* (20), 213-223.
- Zmud, R. W. (1979) Individual differences and MIS success: a review of the empirical literature. *Management Science*, 25(10), 966-979.

Chapter 9

Constructive Design Environments: Implementing End-User Systems Development

John G. Gammack
Murdoch University, Western Australia

The philosophy of end user design proposes an approach to information systems provision where those involved in the human activity context are central to establishing the relevant requirements for their information systems. In this paper we develop the case for centering definitions and process flows on end users in their active situations. We examine the potential for basing integrated IS development upon the constructive and evolutionary processes in the client context. Provision of enterprise-wide IS design environments in which this approach becomes realistic implies a systemic reappraisal of the role of software engineering methods and their place in IS design. With reference to case studies we consider some organisational characteristics in which evolution of specific information systems can be achieved through provision of such design environments. Representative situations at the level of full application design and customisation, workflow definition and enterprise-wide development are considered.

Implementing enterprise-wide end user development as an approach to information systems provision involves shifting the activities of information definition, processes and flows onto end users in their organisational situation. Although such an approach might at first seem to imply prohibitive

training requirements, lack of quality or incompatibility with other enterprise information systems, this is not necessarily the case. Indeed, it is argued that such a requirement will become inevitable for many types of information systems. Rather than having intermediaries anticipate requirements for specific applications, the provision of design environments supporting end user development is suggested. Establishing the requirements for these environments, in which the evolution of specific information systems within organisations becomes possible, then becomes the emphasis of software engineering in conjunction with related management and cultural practices.

Various levels of systemic activity are involved in implementing information systems, and this is becoming increasingly recognised in computing science. Major systems theorists, such as Boulding (1956), and Checkland (1981) have provided classifications of systems relevant to comprehending organisational activity. Boulding's hierarchy recognises systems at nine increasingly complex levels, from the simply mechanical and deterministic, through biological, human and social, to transcendental. Each level has its proper sciences and forms of meaningful explanation. Checkland distinguishes natural, designed (physical or abstract) and human activity systems. Information systems, as commonly understood in the organisational computing literature are not classified neatly by these hierarchies, and this may be attributed to their essentially hybrid status. In their data (closed) aspect they may (appropriately) be rigorously modelled and objectivised, but once data becomes interpreted and contextualised by humans in unanticipated situations, its status changes to information, and becomes referenced to more open and complex systems with less certain or determinable operations. The art of information systems development lies in accommodating both sets of requirements: the integrity of the mechanical within the flexibility of the social, where both are required.

Classic lifecycle based models of IS development emphasise establishing agreed definitions and procedures at the outset of development, and performing translations or mappings from original specifications. Such approaches have frequently been criticised for failing fully to capture the intended user requirements, and for failing to deliver systems relevant to current user needs within time and budgetary constraints. Recently, Wegner (1995) has provided a mathematical proof of why the standard waterfall methodology is doomed to fail. This is because the process of software development is not a fully constrained, but an interactive system. Wegner's conclusion, however, is not surprising, and the history of numerous failures in computer based information systems development has been widely recognised and reported (e.g. Brooks, (1986), Fortune and Peters (1995),

Neumann (1995), Lyu (1995)). Sauer (1993) has provided a qualitative analysis based on case studies, and the continuing tradition of IS failures has historically led to specific methodological adaptations and advances. A recent general thrust of these approaches has been to pay more attention to the user requirements, on the view that if these are not adequately understood, no amount of effort on subsequent program design stages will help.

Rapid Application Development (Martin, 1991), Dynamic Systems Development Method (DSDM, 1996), the spiral model (Boehm, 1988) and prototyping approaches (e.g., Connell and Shafer, 1995) have provided development approaches which address some of these issues, whilst retaining an essential commitment to a structured development by analyst/programmers. Other approaches explicitly involve the end-users to a greater or lesser extent in thoroughly defining or determining the requirements prior to coding. Among these for example, are initiatives in the field of participatory design (Greenbaum and Kyng (1991); Schuler and Namioka (1993); Muller and Kuhn (1993)), soft systems methodology (Checkland 1981), and client-led design (Stowell and West, 1994), all of which recognise the critical contribution of end-user ownership in system specification and its impact on eventual successful uptake. Such insights, however, are predicated on existing models of IS development, and may not be sufficient in themselves to overcome the essential causes of failure.

Our suggestion is that there are other causes of failure in systems development which are not simply amenable to methodological refinements or to rebalanced lifecycle emphases. In addition to the need for clear software specifications, at the level of the emergent properties of systems in action, there are requirements which cannot trivially be predicted by designers. These requirements lie outside the scope of methodology, and are determined by the nature of organisational change, by human information construction and in situated knowledge use and practice. The environment of any system is impacted by other systems, which themselves are in dynamic flux. In turbulent organisational environments where the referents of information systems specifications are subject to change, reinterpretation or reengineering, a deemphasising of the rigid definitions and fixed flows traditionally associated with many IS developments is required. In their stead comes an emphasis on designing enabling policy and technological structures, designing environments for empowered end users, adaptable reorganisation and loose coupling of generic components, from which specific developments can evolve both rapidly and suitably. This is particularly so for the more advanced classes of integrated information and “knowledge creating” systems designed to facili-

tate flexible use of corporate data which will be required by knowledge workers, such as decision support systems, EIS and MIS generally.

Winograd (Winograd and Flores, 1987; Winograd 1995) has argued that “the most successful designs are not those that try to fully model the domain in which they operate, but those that are ‘in alignment’ with the fundamental structure of that domain, and that allow for modification and evolution” (Winograd and Flores, 1987, p 53). The concept of design environments is an extension of this idea, and includes not only software design, but organisation design.

Another argument comes from the global director of change management at consultants KPMG (Jeans, 1996) who has described ‘networked individuals’, highly dependent on, and able to access knowledge as the key to future organisations. In his view, future organisations must develop flexible systems which share knowledge, rather than just process data. Networked individuals will work in “less structured environments, develop abilities to spot trends, and set up knowledge based systems” as companies operate globally, offering staff flexible employment contracts. Such a scenario implies end users playing a greater role in identifying the value in available information, and bringing it into their work. In volatile areas such as arbitrage risk assessment, it is unlikely that intermediaries will either know enough about the business, nor be able to provide relevant information systems in the time scale required.

The need for IS development approaches which can work in contexts of dynamic flux is clear when the contemporary organisational environment is considered. An ethos of project based teamworking, process reengineering, greater customer responsiveness and global competition combines with more educated workforces, layering of organisations and evolving job specifications requiring continuing skills upgrading. Furthermore, in large or distributed organisations, with cohesive departments, there is likely to be more than one culture (Bødker and Pedersen, 1994). Since the dynamics of teams and microcultures are bound to be determined by or emerge from, the component members, it is unwise to over-design from a particular set of views when team composition and purposes may not be stable over time.

New possibilities enabled through Internet-based working, groupware and other shared applications, enable policies of distributed and cooperative working across regions and sites to be planned as technologies and organisational processes integrate. In such environments information has transient value, is highly contextualised and referenced to specific team cultures. With the increase in cross functional or project oriented teams and widescale moves towards market determined service industries, the concept

of the temporary organisation (Lundin 1995) begins to impact on the practice of IS development.

We thus suggest that the advances in clarifying the specification of the software aspect of information systems can usefully be complemented by making provision for the way in which information is constructed and actually used in organisations. Although there are plenty of techniques to ensure data integrity and modeling efficiency, data only becomes meaningful information through interpretation in its active context. Since end users do not always have predictable purposes for data, and economic, legislative or other political contexts can change over the lifetime of a system, we consider the design consequences of these aspects are best delegated to situated end users. This in turn implies a role for IS professionals and organisational software engineers in facilitating the user design, customisation and general enhanceability of systems through appropriate meta-design. A more detailed examination of some of these themes is described in Crowe, Beeby and Gammack (1996): here we extend the general theoretical motivation for the approach, along with specific examples and case studies.

END-USERS AS DESIGNERS

In the film “Good Morning, Vietnam,” the star, Robin Williams, was minimally scripted, and instead blank spaces were left so that his improvisations as a disk jockey could be filmed as they happened. Similarly, in “Kansas City” (using a technique employed in several of his other films), the director Robert Altman allowed Harry Belafonte freedom to make up the lines of his character. Belafonte said he wanted to create a rare, authentic black character, pointing out that the black dialogue needed some inner dimensions that most black roles never get. Though anecdotal, these popular culture examples illustrate a constrained freedom to design details, which acts as a metaphor for the approach to development described in this section.

Bell comments on the relationship between user, developer and context: “Fundamentally, the potential or actual user of the system should be regarded as the expert. Computer professionals can attempt to outline and install systems which will work in various contexts but ultimately users know their context best.” (Bell, 1992, p 51).

The view that end users are the most appropriately placed to establish the norms, conventions, language and microculture in which an IS development emerges has become increasingly enshrined in participatory development practices. This principle is summed up in the phrase “user-as-designer”

discussed in the literature on human-centredness (e.g., Gill, 1991; 1998) and recently elaborated by Lytje (1997). Relocating decision making towards the periphery of organisations allows faster and more contextually tailored responses, as well as the potential to respond to specifically localised emergencies. Placing users in centrally active roles in information systems design has spawned numerous methods in recent use by companies (Presence, 1998). But there remain fears around end user development, that have to do with the quality and accuracy of systems, the training required and the integration of developments with other initiatives and policies. Panko (1997) has compiled a repository summarising research findings on the extent of errors in spreadsheets, and some other end user developments. Inadequate requirements analyses, idiosyncratic customisations, irresponsible usages, worries about scope and complexity, and undocumented and non-transferable developments are other familiar arguments against an unconsidered empowerment of end users. Generally these fears may be subsumed under a sense of loss of the integrity of management control, and a fragmentation of the corporate vision.

Despite such concerns, an increasingly computer literate workforce, advances in useability, the advent of empowering end user tools (e.g., Shah and Lawrence, 1996), better management practices including controls on security and data access, and increasing levels or layers of application interfaces can all be expected to continue as growth areas. Coupled with greater interoperability, ubiquitous networks and protocols for seamless and mobile information interchange, the corporate infrastructure for supporting user based developments is increasingly in place. The UK Department of Trade and Industry's investigation into user-enhanceable systems (DTI, 1993) highlighted many of these issues along with a recognition of the inevitability of such systems in future, and industry analysts have predicted that end users will be doing the majority of future application developments (Gartner Group, 1996).

Leaving aside the above issues, which primarily concern the technological infrastructure to support information use, here we address the dynamic informational environment in which organisational activity takes place, not its data substrate. The view of information adopted here is not one of disembodied scientific objects which can be captured in some canonical structure, but one in which information gains its meaning from practice, and in its contextual situation of use. The anthropocentric or human-centred tradition of systems development has long recognised this. The development emphasis is on providing the infrastructure in which such systems can emerge, relate to current purposes, adapt, or dissolve, with minimal overhead on third

party intermediaries attempting to specify rigid and context bound definitions for each specific circumstance. This allows in-group understandings, communication conventions and other social norms to develop and gain acceptance. Self-organising newsgroups with negotiated discussion threads, and indeed the Internet itself in many other ways provide an example of how a separation of supporting infrastructure from its uses can operate.

Nonetheless, when a top down IS planning strategy is aligned with a longer term corporate view, it is possible to identify directions for application developments which exist within larger systems environments. Scaling up end user development particularly requires ensuring alignment with those areas identified as of corporate strategic interest. Localised initiatives, responding to emergent opportunities or transient problems, are characteristic of end user developments. A challenging role for IS designers, but one viewed as increasingly necessary, is in providing the software development context in which these are integrated within a coherent corporate vision. In such situations, it becomes more appropriate to consider how to provide an environment in which the end users, or team workers, are empowered to determine and construct their own systems of working and information exchange through appropriate managerial and technological provision. There are other, more fundamental, reasons why providing such environments makes sense however, and these reasons are to do with the nature of interpretation and information construction, both at individual and social levels. In the next section, some implications of this view of information systems development are examined.

A CONSTRUCTIVIST VIEW OF IS DEVELOPMENT

The concept of information itself is theoretically problematic. Information is clearly not data, and information systems are not databases. Although such remarks are now uncontroversial, the legacy of their formerly perceived equivalence within computer science remains, and this theoretical confusion still prevents many software developments from achieving their most useful potential. The distinguished MIS professor Jacek Kryt (e.g., Kryt, 1995, 1996) has examined the concept of information in information systems, and contributed vigorously on this theme to internet discussion lists. His summary of the essential distinction captures the idea that information exists in the cognitive realm of the human mind, constructed (rather than represented) by mind, and without that involvement, remains as mere data.

Dervin and Nilan (1986), in an influential paper have elaborated a similar position by contrasting the traditional view of information as essen-

tially an objective property of data or matter, with a constructivist approach focussed on the user. In their view users are seen as “beings who are constantly constructing ... who are free (within system constraints) to create from systems and situations whatever they choose” (p 16).

Such constraints include the perceptual and the sociocultural, and comprise a larger host system or environment within which forms of information are constructed, and to which their meanings are referenced. The biology of such processes, and their relation to linguistically symbolised environments has been seminally described in Maturana and Varela (1980), and, with an emphasis on perception and cognition, in Varela, Thomson and Rosch (1991). Von Foerster (1984) has postulated that the nervous system is organised (or organises itself) to compute a stable reality, and this idea can be extended beyond the individual mind.

Similar stability seeking processes exist in higher order systems, and concepts of learning and knowledge can be applied to suprapersonal systems such as organisations. Choo (1996, 1997) looks at the ways information is strategically processed and used within organisations, and identifies how this determines their capability to stabilise, to grow and to adapt. Organizational knowing is seen as the “emergent property of the network of information-use processes through which the organization constructs shared meanings about its actions and identity; discovers, shares, and applies new knowledge; and initiates patterns of action through search, evaluation, and selection of alternatives.” (Choo, 1997). In particular, several processes of sensemaking are described by which organisations interpret the environment, and form conceptual models as a basis for decision and action. Managers respond to changing or equivocal perceptions of the external environment enacting an organisational environment in which they literally construct some system of constraints, which are then subject to selection processes. Processes of overlaying interpretive relationships then occur to “make sense” of a situation. Inasmuch as such stabilised interpretations prove successful, they are then retained as a conceptual basis for future actions. This views organisations as interpretation systems which exist to produce such stable interpretations of equivocal data about their environment (Weick, 1995).

Such conceptions of information as interpretive sensemaking have been extensively explored in the theoretical literature on autopoiesis and enactive cognition (Varela, Thomson and Rosch, 1991). In this view, purposive activity is considered not so much as stemming from pre-existing representations corresponding to prescribed events, but as an enactment of a world from a historical basis of developed capabilities. Organisational data has an historical provenance, but is contextualised and combined with other data and

information sourced from presently experienced environments. This activity is one of human perceptual and social construction, and transcends meanings simply derivable from data itself. Increasing the specification and precision of data, whether in textual or visually symbolic forms, cannot fully prescribe this activity; a recognition which has inspired conceptions of information systems in semiotic terms (Stamper, 1973; Bøgh Andersen, 1993).

The equivocality of potential interpretations of the environment implies that definitive formulations based on selection and exclusions are likely to be contentious for non-trivial situations, particularly when a variety of stakeholder viewpoints are involved. The selective disambiguation required by precise logical modelling processes, and the typical lack of consensus when conflicting values are involved implies a wicked relationship between specific interpretations and coerced solutions.

Rittel and Webber (1973) originally identified the concept of the wicked problem, which often characterises the situations in which IS solutions must operate. A wicked problem can be described as satisfying the following criteria (paraphrased from Conklin and Weil, (1998):

The problem is an evolving set of interlocking issues and constraints, and there is no definitive statement of the problem. You don't understand the problem until you have developed a solution. There are many stakeholders involved, so the problem solving process is fundamentally social. Getting the right answer is less important than having stakeholders accept whatever solution emerges. Constraints on the solution, (e.g., resources, political ramifications) change over time, ultimately, because we live in a rapidly changing world. Operationally, they change because many are generated by the stakeholders, who come and go, change their minds, fail to communicate, or otherwise change the rules by which the problem must be solved. With no definitive problem, there is no definitive solution. Finally, the problem-solving process ends when you run out of time, money, energy, or some other resource, not when some perfect solution emerges (Conklin and Weil, 1998).

Given this view of the wicked nature of many organisational situations, the prospects for information systems development may seem bleak. In the remainder of this paper, some indicative examples of information systems that take cognisance of these factors are described.

DESIGNING FOR END USER SYSTEMS CONSTRUCTION

Clarke (1997) has suggested that instead of solving wicked problems, it is more appropriate to design solution frameworks, including IT, organisational design and process elements which accommodate the input of many different people and knowledge from multiple sources. Such frameworks allow rapid processing of information by those involved in the problem situation, the development and evaluation of a solution, and further iterations of this process until a temporarily adequate resolution is found. Drawing analogies with Boehm's spiral model, Clarke has designed a set of Lotus Notes™ based tools to provide such a framework for project teams at WL Gore.

More explicitly concerned with software development, are methodologies which take wicked problems and a recognition of the limitations of waterfall developments as their starting point. SCRUM for example (Schwaber, 1996) is a development process for managing object-oriented projects. Based on complexity theory, SCRUM has been designed to "empower small teams to rapidly evolve new and legacy software systems while solving "wicked" problems." (Sutherland, 1998).

Built into such approaches is an emphasis on iterative and evolutionary solutions in which information is adapted by users in their context. There is no attempt to fix a correct definition at the outset of design, instead, users are supported in their construction of solutions by IT, in Gore's case, using a groupware product.

Various other products add end-user value to conventional organisational software, through customisation. Malinowski and Nakakoji (1995) for example have described an approach to end user customisation of interfaces which embodies an ongoing negotiation between the current preferences of those users and the designer's rationale for the development model. Here the choices and rationale for those choices made by the designer are incorporated in an agent which can critique user selected changes, and provide explanations. Users can then decide at any time which changes to the interface they wish to implement.

McCartney (1996) has also noted the impact of context and dynamic change on communication and computation requirements, arguing for the empowerment of end users to make their own customized applications. He provides various mechanisms enabling end user construction of distributed multimedia applications, designed using the Programmer's Playground (Goldman et al., 1995), a software library and system which allows such end user construction from modular software building blocks.

Elsewhere Bridger Systems (1997) have developed the ORBIT database system, which is claimed to significant advantages over traditional database systems including:

- Automatic Compatibility - Tables, records, and fields can be added or removed without changing any existing data. This is considered of great benefit in systems that change or grow rapidly.
- End User Enhanceability - Applications can be developed in ORBIT allowing the end user to add custom fields, records and tables. These custom areas are maintained even when a new version of the application is delivered.

Like ORBIT, the IDIOMS decision support design environment (Gammack, Fogarty, Ireson, Battle and Cui, 1992) embodied this latter principle, in which end users could build predictive business models from the corporate database, but also customise those with contextual knowledge, subjective judgements and new constraints affecting data interpretation. This design environment was application neutral, and allowed specific decision support models and automated classification systems to be rapidly and easily designed by end users. A combination of managerial and technical procedures allowed various levels of end user control of model design to be implemented.

The approach to establishing the requirements for developing this design environment was participative, using modified JAD sessions (Wood and Silver, 1989). Broadly, this entailed determining the outline user requirements at a high level, addressing problems of strategy, direction or infrastructure, and concurrently making efforts to understand the nature of what users at operational levels required to support their practical activities. Ethnographic approaches (Hammersley and Atkinson, 1995), where feasible, are seen as well suited to this aspect of the work. The approach however specifically did not require the development team to understand any application in detail, nor how business decisions were actually made. Consequently, detailed diagrams of processes and flows were not required. Having established the client's general requirements, effort was concentrated on providing a software infrastructure at that level, within which specific application developments could occur, with several exemplar applications seen through to implementation, again with no detailed process or entity modelling required. The difference in this approach over some traditional methods lies in the provision of a means whereby the lesser changes in the organisation may be accommodated, through designing technological or organisational solutions that meet a greater strategic requirement.

This development approach was also adopted in another project involving organisational change, both at the technological and at the cultural level.

(Gammack and Jenkins, 1998) The client was a large marine engineering firm distributed over several sites across the UK. Large contracts implied that teams distributed over these sites were required to work together, and various problems with this had been generally identified. These included the disruption due to temporary co-location of teams; the requirement for senior managers to be (often for brief inputs only) in various places “at once,” and the time delays involved with the dependencies of sequential working. (Turner and Turner, 1994). Moreover distinct cultures of working existed at each site, due to historical and other factors. Computer Supported Collaborative Working using groupware environments was seen strategically as offering direct benefit in these and other areas, and motivated the detailed project.

In this project, it was considered less important to build a new tool addressing specific problems than to design an environment in which such generic problems would be minimised. Thus, numerous existing proprietary technologies, which the company either already had in place or which could be readily purchased, were identified as being relevant during a more detailed requirements analysis phase. These included both internal and external electronic mail, a shared electronic whiteboard, remote application sharing, telephone conferencing and desktop videoconferencing. The major software engineering effort was required to link these together appropriately and to develop customised add-ons specific to the organisational activity of collaborative engineering design.

With these considerations in mind, the concept of a shared electronic daybook, or on-line design journal was proposed, and prototyped (Gammack and Jenkins, 1995). This essentially hosted the existing applications (sketchers, spreadsheets, word processors, CAD applications and so on) that designers actually used in their work but distributed its accessibility across emergent and delegated project teams, with abstracting capabilities for corporate level recording of activities. This, augmented with a design history editor made provision for reuse of designs, avoiding repeating previous failures and general organisational learning. If it is the case that organisations can implement processes to identify stable interpretations, recurrent patterns and successful solutions, and record that history as a basis for future action, an elementary form of learning can take place.

Several initiatives on the managerial side also helped to create an environment of cooperative development. These included: setting up a fast and secure network between sites, widening the access culture of e-mail and web communication, providing relevant training, non-coercive introduction

of the technologies, and effectively publicising the benefits of this form of work.

The motivation of (particularly) younger staff to upgrade skills and expertise for career development, along with other widely apparent benefits at operational level helped those involved in the design activities to organise their own ways of working without prescribing any specific method. Simply achievable outcomes such as enabling access to one's own files and to keep in touch with colleagues at the home site when on remote location, were immediately appreciable benefits. Others, such as improving routine communications across sites (avoiding telephone tag; being able to exchange soft copies of work rather than fax printouts and so on) were also widely apparent advantages. Again, this is an example of designing an information environment in which end users could construct their own solutions to generic problems, without intermediaries prescribing or unnecessarily limiting those solutions, and paying attention both to the design of the software infrastructure and of the organisational behaviours.

This latter example begins to go beyond an engineering focus of software design towards designing for workflows and information flows at the level of human activity. Buckingham (1997) has examined such principles in his analysis of the "unorganised organisation." The activity of downstructuring is seen as not so much about reducing the number of employees, but rather freeing employees for creative knowledge work through eliminating systems and procedures including job titles and paper-based administrative processes. Such procedures are viewed as necessary because in a complicated and, in Buckingham's term, unorganised, world traditional managerial activities of understanding and controlling events and people are harder to apply. Instead managers must "(let) strategies emerge, make reversible commitments and keep decision making vague." Successful companies applying these principles are beginning to emerge and become recognised, as the following examples show.

Although many organisations can give examples of empowering teams to redesign their work, perhaps the concept of temporary, self-organising workforces is epitomised by the case of Oticon, a company with a long history of humanistic and user centred practices. Oticon is a Danish hearing aid producer (the world's largest) whose organisation is so fluid that employees have no permanent office space, and all jobs are carried out in project groups with no managers in any traditional sense. Instead, project managers (anyone with good ideas) have a prime function of motivating employees to join their project. (Bjorn-Andersen and Turner, 1994). Oticon's CEO, in regaining the number one position the company had lost, implemented a policy that has

been described as deliberate disorganisation, (Labarre, 1996) with his view that in future organisations “staff would be liberated to grow, personally and professionally, and to become more creative, action-oriented, and efficient.” Their information systems are predicated on the policy that face-to-face communication is the most important, and that by moving around and mixing or chance encounters with workers in other specialisms, both a greater understanding of what people do, and fewer perceptions of enmity arise.

Technologies, particularly groupware, support these processes, but the primary factor in project success is the skills of the team, not especially documents from the past. (Sellen and Harper, 1997). Documents are seen as being primarily relevant to current projects, and of very limited subsequent value. The following example of an IS development within Oticon illustrates an effective balance between unconstrained end user development freedom, and planning by IS professionals.

When an electronic filing system was designed in Oticon, the first approach involved IS planners asking employees what kinds of data they would like access to, but this soon became too complex. Then a very flexible system was provided, allowing everyone to design their own organisation of files, which did not work, overestimating people’s ability to do this. Finally IT managers provided a basic category structure, corresponding to project phases and alterable only by project leaders. Users were encouraged to use keywords and sensible file names, and soon fell into good habits. Access rights were abandoned when (in this case) their relative inutility was recognised. This approach proved successful, and Oticon continue to pioneer innovative work practices.

In Kao’s (Kurzmann, 1996) description, Oticon “changed from sheet music to jazz ... It changed the organization from being built around a traditional flow of processes to one that was structured around a bag of projects and new ideas.” Those ideas originate with the end users, and drive the dynamic use and development of supporting technologies, a policy exemplified in the next example.

In Colruyt, the leading Belgian food discount chain, IS are developed only by people who have worked on the shop floor for two years, and IS developments are bottom up and end-user led. For example, checkout staff noticed that the existing system of scanning multiple product units was inadequate. Some products such as milk were sold in shrink wrapped units of four cartons, and the system of scanning cartons once, then typing the multiplier (4) was perceived as inefficient and error prone, as staff sometimes forgot to type the multiplier. Affected staff across stores met, and reached agreement that a new dedicated barcode would be better. They then devised

a proposal, with costings, projected time and money savings and after discussion on software development aspects with the IS department made the proposal to top management, who allocated the resources readily. Millions of francs have been saved since. (Janson and Taillieu, 1996).

These latter initiatives pervade information systems developments across a whole company, and to greater or lesser extents the end users, perhaps in conjunction with information systems professionals determine their requirements. The potential of open intranet, extranet or Internet-based workplace environments however enables enterprise wide integration of communicative activity and knowledge work, with a limited level of end user application specific expertise required. Databases can be searched, remote applications launched, forms filled and so on to deliver in, and transmit out relevant information across knowledge workers.

Superstructures can be designed around this basis to create any specific environment required by an enterprise. For example Neilsen and Sano (1994) designed the Sun Microsystems internal web site to have a consistent interface for easier use. A few usability experiments established the user requirements, which were then established across the organisation's internal network.

Other approaches take advantage of actual practices to determine norms: and various ethnographic methodologies may be applied here, such as unobtrusive measurements. An archetypal example of this occurred in the design of a new university campus, where no paths were planned and only grass was laid in the grounds. After some time, aerial photographs clearly showed the paths trodden by students, and these were paved, on the assumption that they represented the most efficient routes between popular locations. Analogously, Hill et al (1994) have examined issues of information filtering at the social or community level, with particular regard to such history of use information. Marking visited hypermedia links, using page counters, recording file editing frequencies and so on all provide information in an unobtrusive way in which user needs and preferences can be established, and if necessary, formalised as *de facto* procedures, particularly in administrative matters (HEIRA, 1994).

Opinion is divided on the value of such practices, which requires more research. The admonition not to "pave the cowpaths," (Hammer and Champy, 1993) addresses the futility of embalming possibly inefficient or exploratory practices, as the software engineer Robert Early's (1997) analysis of the Irish boreen, or minor road, has suggested. The example of Oticon above also suggests that a balance between planning and spontaneity is required.

CONCLUSIONS AND FURTHER RESEARCH

So far we have discussed the provision of information systems design environments with examples at the level of application customisation, application design, workflow design, and enterprise-wide development. Many of these examples are as yet more suggestive than paradigmatic, and several issues requiring more detailed investigation remain. These include appropriate training and educational development, relations between tools and social processes, stakeholder conflict resolution, group design activities, appropriate roles for analyst/facilitators and validation concerns.

Going beyond even these concerns, is the ethical design of IS development environments which foreground culturally appropriate and indigenous constructions of information. Trees and Turk (in press) have researched the design of a multimedia repository for cultural heritage information of the Ngaluma, Injibandi and Banjima peoples of Ieramugadu (Roebourne) in Western Australia. In this work, the informational elements, and the narrative structures imposed on those elements are designed and assembled by community members themselves to suit any specific set of requirements, whether those be in educational, negotiation or reconciliation settings. This highly participative approach has involved extensive building of rapport with the community, and identifying “information needs, sources, narrative structures and possible information use scenarios” along with multimedia prototypes for specific requirements. The technologies and associated training have been seen by the local community as empowering, and the work relates to a wider context of cultural studies and legal research projects (Turk 1996, Turk and Mackaness, 1995).

This approach to indigenously constructed, “content flexible” design explicitly addresses the issue of the appropriate use of secret or sacred information. Such information can be compromised by a displaced embodiment in any specific interpretation for an externally designed application, and this authenticity applies to many community information system developments (e.g., Beeson and Miskelly, 1998). This type of work, in which the concept of end user empowerment is coherently extended from the technical through to the cultural levels of systemic activity is seen as a promising direction for future research.

ACKNOWLEDGMENTS:

I would like to thank Richard Beeby and Andrew Turk for helpful comments on this paper.

REFERENCES

- Beeson, I. & Miskelly, C. (1998). Tactical confabulation. In: S. Probert & R. Stephens (Eds) *PAIS II: Proceedings of the Second International Symposium and Workshop on Philosophical Aspects of Information Systems*, University of the West of England, Bristol UK.
- Bell, S. (1992). Information systems and developing countries *Computer Bulletin*, June/July.
- Bjorn-Andersen, N., & Turner, J. (1994). Creating the twenty-first century organisation: the metamorphosis of Oticon. In R. Baskerville, S. Smithson, O. Ngwenyama, & J.I. DeGross (eds.), (Eds.) *Transforming organisations with information technology*, (pp379-393), London: North-Holland.
- Bødker, K., & Pedersen, J.S., (1991). Workplace Cultures: Looking at Artifacts, Symbols and Practices. In J Greenbaum & M. Kyng (Eds.). *Design at Work: Cooperative Design of Computer Systems*. (pp121-136) Hillsdale NJ: Lawrence Erlbaum Associates.
- Boehm, B. W., (1988). A spiral model of software development and enhancement *IEEE computer* 21 (5) 61-72.
- Bøgh Andersen, P. (1990). *A theory of computer semiotics: semiotic approaches to construction and assessment of computer systems*. Cambridge: Cambridge University Press.
- Boulding, K.E. (1956). General systems theory - the skeleton of science, *Management Science*, 2, 197-208
- Bridger Systems (1997). <http://bridger.link-usa.com/brhm25.htm> (August 10, 1998)
- Buckingham, S. (1997). Unorganization: the company handbook Electronic book available at <http://www.unorg.com/company.htm> (August 10, 1998)
- Checkland, P.B. (1981). *Systems thinking, systems practice* Chichester: Wiley.
- Choo C.W. (1996). The knowing organization: how organizations use information to construct meaning, create knowledge, and make decisions *International Journal of Information Management*, 16(5), 329-340.
- Choo C.W. (1997). *The knowing organization: How organizations use information to construct meaning, create knowledge, and make decisions*. Oxford: OUP.
- Clarke D (1997). Hurray for real time management! (response) Available at URL <http://www.learning-org.com/97.05/0048.html> (August 10, 1998)
- Conklin, J, & Weil, W. (1998). Wicked problems: Naming the pain in organizations. Available at URL <http://www.gdss.com/wicked.htm> (August 10, 1998)

- Connell, J. L. & Shafer, L. B. (1995). *Object-oriented rapid prototyping*. Prentice Hall/Yourdon Press.
- Crowe, M., Beeby, R., & Gammack, J. (1996). *Constructing systems and information: A process view*. Maidenhead: McGraw Hill.
- Dervin, B., & Nilan, M. (1986). Information needs and uses. In M. E. Williams (Ed.), *Annual review of information science and technology* (vol. 21) , 3-34, White Plains, NY: American Society for Information Science.
- DSDM (1996). Dynamic systems development method, Ashford, Kent: Author. <http://www.dsdm.org> (August 10, 1998)
- DTI (1993). *Department of Trade and Industry Report on user enhanceable systems*. London: Author.
- Early, R. (1997). Road etiquette in Ireland. Available at <http://www.csn.ul.ie/~robert/road/road.html> (August 10, 1998)
- Fortune, J., & Peters, G. (1995). *Learning from failure: the systems approach*. New York: Wiley.
- Gammack J.G., Fogarty, T.C., Battle, S.A., Ireson, N.S., & Cui, J., (1992). Human centred decision support: The IDIOMS project. *AI & Society* (6), 352-366
- Gammack, J.G., & Jenkins D.G. (1995) The Online design journal: retrospect and prospect, *Computing and Information Systems* 3 (1).
- Gammack, J.G., & Jenkins D. G. (1998) Collaborative design in distributed organisations: requirements and evaluation, *Proceedings of the Ninth Australasian Conference on Information Systems*, Sydney, vol. 1, 241-252..
- Gartner Group (1996, May). PC user skills to double. *Computer Consultant*.
- Gill, K.S. (1991). The human centred tradition in Europe, *Systemist* 13 (1 and 2)
- Gill, K.S. (1998). Human centred systems <http://www.it.bton.ac.uk/research/seake/home.html> (August 10th 1998)
- Goldman, K.J., Swaminathan, B., McCartney, T.P., Anderson, M.D. & Sethuraman R. (1995). The programmers' playground: I/O abstraction for user-configurable distributed applications. *IEEE Transactions on Software Engineering*, 21(9), 735-746.
- Greenbaum, J. & Kyng, M. (1991). *Design at work: Cooperative design of computer systems*, Hillsdale, New Jersey: Lawrence Erlbaum.
- Hammer, M, & Champy, J. (1993). *Reengineering the corporation*. New York: Harper Collins Publishing, Inc.
- Hammersley, M. & Atkinson, P. (1995). *Ethnography* (2nd edition) London: Routledge.

- HEIRA (1994). *What presidents need to know about the payoff on the information technology investment*, HEIRAlliance Executive Strategies Report #4, University of Colorado: Author.
- Hill, W., Rosenstein, M., & Stead, L. (1994). Community and history-of-use navigation Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web Available at: <http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/hill/home-page.html> (August 10th 1998)
- Jansen, M.A. & Taillieu, T. (1996). Communicative action theory is profound, but is it fecund: Experiences of a Belgian discount chain. In S. Wrycza & J. Zupancic (Eds.) *Proceedings of the 5th International Conference on IS Development* (459-486) Gdansk: Gdansk University Press.
- Jeans, M. (1996, September). Systems key in networked individual age. *Computing*, 12/9/1996
- Kao, J. (1996). *Jamming: the art and discipline of business creativity*. Harper Business Books
- Kryt, J. (1995, November). Information highway or commercial data dump? Pigulki, 20, Available at <http://www.waw.pdi.net/lynx/server/Pigulki/issue20/p20kryt.html> (August 10, 1998)
- Kryt, J. (1996). All that glitters on the internet is not information In S. Wrycza & J. Zupancic (Eds.) *Proceedings of the 5th International Conference on IS Development* (121-134), Gdansk: Gdansk University Press.
- Kurtzman, J. (1996). An interview with John Kao. *Journal of Strategy and Business*, Fourth Quarter
- Labarre, P (1996). *This organization is dis-organization*. Fast Company 3.
- Lundin, R. A. (1995). Editorial: temporary organisations and project management. *Scandinavian Journal of Management* 11 (4) 315-318
- Lytje, I. (1997). Software as multimedia text: Design of open multimedia systems. *AI&Society*. Available at <http://www.hum.auc.dk/~inger/soft.htm> (August 10, 1998)
- Lyu, M.R. (1995). *Handbook of software reliability engineering*. IEEE Computer Society Press / McGraw-Hill.
- Malinowski, U. & Nakakoji, K. (1995) Using computational critics to facilitate long-term collaboration in user interface design. *Human Factors in Computing Systems*, CHI'95 Conference Proceedings, ACM, pp. 385-392.
- Martin, J. (1991). *Rapid application development*. New York: MacMillan Publishing.

- Maturana, H.R. & Varela, F.J. (1980). *Autopoiesis: The realisation of the living*. Dordrecht:Reidel.
- McCartney, T. P. (1996). *End-user construction and configuration of distributed multimedia applications*. D.Sc. Thesis, Washington University.
- Muller M J & Kuhn S. (1993). Introduction to special issue on participatory design. *Communications of the ACM*, 36 (4).
- Neilsen J. & Sano D. (1994). SunWeb: User interface design for sun microsystem's internal web. In: *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*. Available at: <http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/nielsen/sunweb.html> (August 10th 1998).
- Neumann, P.G. (1995). *Computer-related risks*, Addison-Wesley / ACM Press.
- Panko, R. (1997). Spreadsheet research repository (SSR) <http://www.cba.hawaii.edu/panko/ssr> (August 10, 1998)
- Presence (1998). The Presence forum <http://www.presenceweb.org/methodslab/body.html> (August 10, 1998).
- Rittel, H., & Webber, M. (1973). Dilemmas in a general theory of planning. *Policy Sciences*, 4, 155-169 Amsterdam: Elsevier.
- Sauer, C. (1993). *Why information systems fail: a case study approach*. Henley on Thames: Alfred Waller.
- Schuler, D., & Namioka, A. (Eds.) (1993). *Participatory design: Principles and Practice*. Hillsdale NJ: LEA.
- Schwaber, K. (1996). Controlled chaos: living on the edge. *American Programmer*, April.
- Sellen, A.J., & Harper, R.H.R. (1997). Oticon: Experiences of a Paperless Office. Unpublished Report, Rank Xerox Research Centre, Cambridge.
- Shah, H.U., & Lawrence, D.R. (1996). A study of end user computing and the provision of tool support to advance end user empowerment.. *Journal of End User Computing* 8 (1), 13-21
- Stamper R. (1973) *Information in business and administrative Systems*. London: Batsford.
- Stowell, F.A., & West, D. (1994). *Client-led design*. Maidenhead, McGraw Hill.
- Sutherland, J. (1997). <http://www.jeffsutherland.org/scrum> (August 10th 1998)
- Trees, K., & Turk, A.G., (in press). Reconciling space. In R. Barcan & I. Buchanan, (Eds.), *Spaciographies: Essays in Australian space*. Sydney: University of Sydney Press.

- Turk, A. G., (1996). Presenting Aboriginal knowledge: Using technology to progress native title claims. *Alternative Law Journal*, 21(1), 6-9
- Turk, A.G., & Mackaness, W.A. (1995). Design considerations for spatial information systems and maps to support native title negotiation and arbitration. *Cartography*, 24(2), 17-28.
- Turner, S.E., & Turner, P. (1994). Expectations and experiences of CSCW in an engineering environment., *Collaborative Computing*, 1(4), 237-254.
- Varela F., Thomson E., & Rosch E. (1991). *The embodied mind: Cognitive science and human experience*. Cambridge MA: MIT Press.
- Von Foerster, H., (1984). On constructing a reality. In P. Watzlawick (Ed), *The invented reality*, New York: Norton.
- Wegner, P., (1995). Interactive foundations of object-based programming. *IEEE Computer* 28(10), 70-72.
- Winograd, T (1995) From programming environments to environments for design. *Communications of the ACM*, 38(6), 65-74.
- Winograd, T., & Flores, F. (1987). *Understanding computers and cognition: A new foundation for design*. New Jersey: Ablex.
- Wood, J., & Silver, D. (1989). *Joint application design*. New York: Wiley.

Chapter 10

An Information Systems Design Framework for Facilitating TQM Implementation

Nazim U. Ahmed
Ball State University, USA

Ramarathnam Ravichandran
Design Systems, USA

This paper provides a framework for information systems (IS) design for TQM implementation. The framework consists of three main phases. In the first, TQM implementation tasks are established. These tasks include identifying customer satisfaction variables (CSV), translation of CSV to firm response variables (FRV), benchmarking, and continuous improvement. The second phase includes analyses of communication effectiveness requirements between the organizational entities such as sales/marketing, top management, operations, accounting/finance and also with the customers. In the third phase, appropriate IS component inventories for different communication interfaces are generated. This was accomplished by first mapping the TQM implementation tasks for the communication interfaces. Then appropriate IS/IT solution was recommended for each interface. The final IS design is achieved by integrating IS components at technological, functional, and strategic levels. Finally, a hypothetical example for a large manufacturing firm is provided.

Total Quality Management (TQM) is a philosophy that emphasizes customer satisfaction as a driving force for all organizational activities. It results in many benefits to organizations (Chalk, 1993; Sabbaghi, 1990; Rooney, 1990; Vansina, 1990). Several approaches to TQM have been proposed (Adam, 1994; Flynn, 1994; Caudron, 1993; Powell, 1995). We adopt the definition of TQM from Flynn (1994) "An integrated approach to achieving and sustaining high quality output, focusing on the maintenance and continuous improvement of processes and defect prevention at all levels and in all functions of the organization in order to meet or exceed customer expectations."

A number of studies have discussed TQM implementation processes (Ayres, 1993; Sabbaghi, 1990). Some others tried to relate TQM to operating and financial performance (Adam, 1994). Some of the current research have identified an integrated organizational communication system as a critical success factor for TQM Implementation. For example, Schoenberger (1983) and Tillery (1985) concluded that co-operation, coordination and integration of many different functions within the organization is a key aspect of total quality management. Flynn (1994) describes the necessity of linkages between every pair of functions, and forming a web like networking of all functions. Adam (1994) and Powell (1995) through empirical studies have concluded that factors such as objective feedback on performance, and empowerment are more significant than certain other factors such as process improvement and training.

Several of the above-mentioned studies in the TQM area have established the importance of an integrated organizational communication system. However, there is a lack of theoretical or empirical research to suggest how this can be done.

In a traditional organization, growth of information technology (IT) often is not well planned. Most of the growth in IS/IT occurs in pockets and in isolation (Doll and Vonderembske, 1987). Generally, departmental or individual managers vie for sophisticated IT in their own domain. Most often, a decision to implement such technology is born out of the individual desire to be technologically up-to-date rather than from some business necessity. This is contrary to the TQM strategy. Innovative organizations are relying increasingly on IT for maintaining and sustaining the strategic advantage over their competitors (Ali and Miltenburg, 1991; Goldhar and Jelnik, 1985; Kettinger, et al., 1994; King and Teo, 1994).

In the last few years, we have seen the explosion of technologies such as the Internet, intranet, extranet, data mining, and data warehousing which have the potential of alleviating some of the pitfalls of traditional culture. Also, the

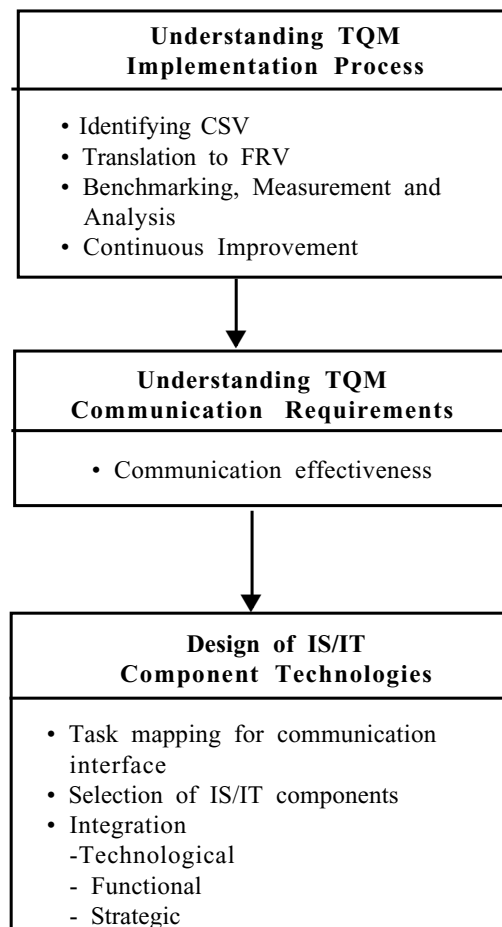
recent issue is not about lack of communication, rather it is about how to make communication effective.

In this paper, we propose an approach for designing and using IS for effective integration of organizational communication to support TQM implementation. The paper is organized as follows. In the next section, we discuss the four tasks for TQM implementation. Effective communication requirements for these tasks are identified in the following section. The next section matches IT for meeting communication effectiveness needs. Then, integration through IS at several levels is outlined. An example scenario of integration of IS is provided. The paper concludes with a discussion of future research directions.

TQM IMPLEMENTATION PROCESS

Figure 1 provides a general framework for IS design for TQM implementation. Four tasks in the TQM implementation process are identified in the figure. A brief discussion of each follows.

Figure 1: Framework for IS design for TQM Implementation



Task 1: Identifying Customer Satisfaction Variables (CSVs)

The ultimate TQM goal is to attain a high level of customer satisfaction (Marquardt, 1992). Customer satisfaction is multi-dimensional in nature (Mathers, 1991; Rooney, 1990; Ross, 1993) consisting of many CSVs. Some are generic; e.g., customers want a well-performing, reliable, affordably priced and long-lasting product. Although the set of CSVs may be constant, the relative importance of each CSV may change over a period of time.

Customer satisfaction is dynamic in nature and is influenced by various factors in the business environment, especially competition. Consider the case of price as a customer satisfaction variable. In the personal computer industry, a customer's perception of price for a given performance level is constantly changing due to competition. As a result, CSVs need to be continuously monitored.

Garvin (1987) discusses eight areas of CSVs. They are 1) performance; 2) features; 3) reliability; 4) conformance; 5) durability; 6) serviceability; 7) aesthetics; and 8) perceived quality. Specific variables may be identified from each of these areas. Pursuing perfection in all of these may prove to be neither prudent nor possible (Garvin, 1987). Firms may choose to pursue only certain of these variables; e.g., Wal-Mart may pursue price and product availability, while L.L. Bean seeks reliability and durability.

A-B-C analysis (Meredith, 1992) can be used to narrow down the set of CSVs that a firm wants to focus on. Category "A" variables which are few in number, will have the greatest impact. The variables in the "B" category are moderately important and should also be considered; however, "C" variables are trivial, and have marginal impact. Table 1 gives some examples of CSVs and their categorization in a hypothetical situation. A-B-C classification is contingent on the nature of the product or service; e.g., appearance may be classified as a "C" for machine tools, but as an "A" for garments.

Task 2: Translating CSV to Firm Response Variables (FRVs)

Customer satisfaction variables generally define customer expectations and needs, often in broad and subjective terms. Sometimes they cannot be objectively measured. In such cases, it is important to map these CSVs to some measurable variables (Garvin, 1987). These measurable variables hereby are called firm response variables (FRV). Organizations need to manage these variables and track their performance over time. Table 1 presents some examples of this mapping process. The mapping can be either one-to-one which implies that a CSV can be represented by one FRV, or one-to-many implying the necessity of using multiple FRVs for a single CSV. The case of many-to-one relationship is not very prevalent. In Table 1 if price is

Table 1: Examples of Mapping of Customer Satisfaction Variables (CSV) to Firm Response Variables (FRV)

Category	CSV	FRV
A	Price	Unit cost Unit overhead cost
	Performance	Miles per gallon (automobile) Mega Hertz (computer)
	Reliability	Number of repairs per time period
B	Service	Time between request and service delivery
	Durability	Average life of the product
C	Appearance	Size (big or small)
	Added feature	Number of different options available

a CSV then corresponding FRV may be unit cost, and unit overhead cost which affect the pricing strategy.

Task 3: Benchmarking and Tracking of FRVs

Once CSVs are mapped to meaningful FRVs, then it is important to set some standards and to design and implement a system of measurement. Benchmarking (Camp, 1993) is the continuous process of measuring products, services and practices against the toughest competitors or those companies recognized as industry leaders. Industry standards, from the world class firms and customer expectations are useful in setting up desired levels of FRVs. In the short run, the standards may reflect such factors as organizational capability, and cost competition. In the long run, the benchmark (Camp, 1989) must reflect the desire of the company to be one of the best in the business. The tracking system should monitor the performance of the FRVs against the benchmark. The measurement system should employ simple and accurate criteria (Chung, 1989; Tobin, 1990). The information provided should be concise, timely and easy to understand. Such information will promote accurate data collection and actual usage.

Task 4: Management of Continuous Improvement Process

One of the major premises of TQM is continuous improvement. Traditionally, U.S. managers maintain product and processes until they can be

replaced by new technology (Evans and Lindsay, 1996). However, the cumulative effect of successive incremental improvements and modifications to established products and processes (Evans and Lindsay, 1996) can be very large and may outpace efforts to achieve technological breakthroughs (Dertouzos et al., 1989).

Continuous improvement cannot be implemented by benchmarking and tracking FRVs alone. It involves a lot of planning and behavioral changes. It necessitates a change in the organizational culture from layered and bureaucratic to responsive and empowered (Levine, 1992; Ludeman, 1992). It needs an organizational communication system and a management system that emphasizes long-term strategy rather than short-term success. An organizational communication system does not necessarily always need to be technology driven. Technology facilitates organizational communication and the management should take advantage of it (Blest et al., 1992; Patten, 1991). However, it is people who will ultimately communicate. So an empowered organizational culture is the ultimate facilitator of effective communication for continuous improvement (Hendricks and Triplett, 1989; Macoby, 1992).

UNDERSTANDING TQM COMMUNICATION EFFECTIVENESS REQUIREMENTS

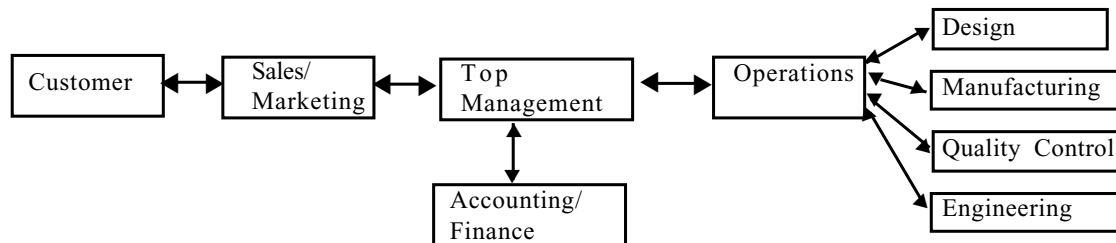
An IS framework for TQM implementation must take into account the nature and specifics of effective TQM communication requirements. In this section the nature of TQM communications requirements is conceptualized. The specifics are situation dependent and may vary between large and small firms, between service and manufacturing. For example, a small service firm like an architectural firm may use face-to-face communication with its customer while an engineer in an automobile manufacturing firm may get feedback through mostly consumer surveys.

Figure 2 presents a comparison between traditional and TQM communication. Traditional communication follows a chain of command type of structure whereby the customer demands and expectations are filtered down through sales and marketing and management to the operations functions. This slow communication may affect competitiveness by delaying product changes and product introduction.

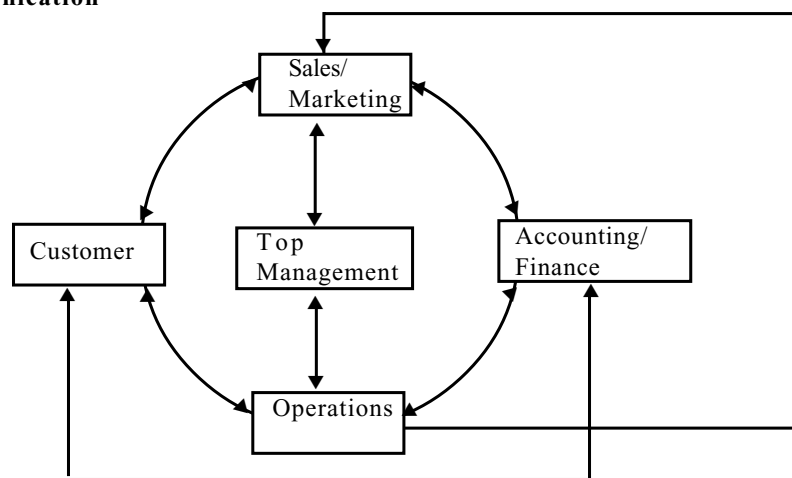
The second diagram in the figure depicts the process of TQM communication. Here the chain of command type of structure is replaced by a less rigid and more empowered structure. Most of the departments interface with the customers. Also, the functional departments interact with each other. It is

Figure 2: Comparison of TQM Communication and Traditional Communication

Traditional Communication



TQM Communication



obvious that the sales/marketing function should have most communication with the customer. However, in a TQM environment all departments should have some form of interfacing with the customers. The specifics of this interface may vary depending on the size and type of organization. For example, the operations department of a large firm may understand the current customer expectations from customer satisfaction database and occasional face-to-face meetings with the customers, while a manager of operations of a small manufacturing firm should interface more frequently with the customers through face-to-face meetings or by phone.

Top management is the hub of all TQM communication. Top management should have frequent communications with all functional areas and also some communications with the customers. The communication culture should be empowered so that; 1) all the functional areas of the organization may provide quick, timely and uninhibited feedback; and 2) the employees may make quick decisions to provide customer service and achieve customer satisfaction. Sometimes the middle level management may view the empowerment process as the shifting of power from them to the lower levels and may

try to resist changes (Conger and Kanungo, 1988; Frey 1993). Top management should manage the empowerment process by earning the trust of middle managers and also involving them in the process. Even though the ultimate responsibility for the tasks delineated in the TQM implementation process is on the top management, they cannot do it alone. Huston (1992) suggested a four-step process. These are; 1) visualization —clear goals; 2) formalization — tightening up the planning and deployment process; 3) individualization — defining clear individual expectation and rewards; and 4) socialization — the need to create shared values and trusting relationships throughout the organization.

Achieving Communication Effectiveness

Currently with the proliferation of communication technologies, the issue is not whether there is communication or not. The main focus of any IS design framework should be on making communication effective for TQM implementation. In an organization, communications can be achieved by a combination of methods. These could be classified into two broad categories: 1) people-intensive; and 2) technology-intensive. Examples of people-intensive communication include face-to-face meetings, phone conversation, voice-mail and memo. Technology-intensive communication methods include electronic data interchange (EDI), local area network(LAN), electronic mail, wide area network(WAN), internet, intranet, extranet, data-mining and data warehousing.

There is evidence that (Frey, 1993;Hart and Schlesinger, 1991; Ludeman, 1992) an empowered and less rigid communication culture has a positive impact on TQM implementation. So it is important that the TQM communication requirements be analyzed from an empowerment focus before the communication interface and technology is designed.

Table 2 presents a communication effectiveness matrix for a hypothetical organization. The entries in the matrix show the required nature of communication interface to achieve effectiveness. For example, there is no communications between the customers. However, the company may think that it is important for satisfied customers to propagate product information to other prospective customers. This can be achieved by designing the customer-to-customer interface, which can be identified as less frequent people-oriented (LF, P) and frequent technology-oriented (F, T). For example, a software firm can arrange users group meeting. They can establish message board and chat room on the Internet.

From Table 2, it can be seen that the communications between the operations and the customer can be made more effective by less frequent

Table 2: Communication Effectiveness Matrix

Internal/ External Entities	Customer	Sales/ Marketing	Top Mgmt.	Operations	Accounting/ Finance
Customer	LF, P F, T	F, P F, T	LF, P LF, T	LF, P F, T	F, T
Sales/Marketing		F, P F, T	F, P F, T	F, P F, T	LF, P F, T
Top Management			F, P F, T	F, P F, T	LF, P F, T
Operations					LF, P F, T
Accounting/ Finance					F, P F, T

LF = Less frequent

F = Frequent

P = People-oriented

T = Technology-oriented

people-oriented (LF, P) and frequent technology-oriented (F, T) interface. This implies that operations personnel need to meet occasionally with the customers face-to-face or talk to them on the phone. However, they need more frequent and structured input through database and computer networks.

IS DESIGN STRATEGY

Effective utilization of IT provides an organization with competitive advantage (Kivajarvi and Saarinen, 1995; Sayeed and Brightman, 1994). TQM implementation necessitates an IS strategy which supports its goals. The strategy for IS design should address the following major issues: 1) task mapping for the communication interface; 2) design of IS/IT component technology; and 3) integration.

Task Mapping for the Communication Interface.

In the previous section, four major tasks for TQM implementation process were delineated. Table 3 shows the mapping of these tasks with the communication interfaces. Table 3 is a modified version of Table 2, whereby

Table 3: Mapping the Implementation Tasks

Internal/ External Entities	Customer	Sales/ Marketing	Top Mgmt.	Operations	Accounting/ Finance
Customer	LF, P F, T <div>1</div>	F, P F, T <div>1,4</div>	LF, P LF, T <div>1,4</div>	LF, P F, T <div>1</div>	F, T <div>3</div>
Sales/Marketing		F, P F, T <div>1</div>	F, P F, T <div>1,2,3,4</div>	F, P F, T <div>1,4</div>	LF, P F, T <div>1,4</div>
Top Management			F, P F, T <div>1,2,3,4</div>	F, P F, T <div>2,3,4</div>	FP F, T <div>3,4</div>
Operations				FP FT <div>1,2,3,4</div>	LF, P F, T <div>3,4</div>
Accounting/ Finance					F, P F, T <div>3,4</div>

1=Identifying customer satisfaction variables

2=Translation to firm response variable

3=Bench marking, measurement and analysis

4=Activating continuous improvements

the implementation tasks are superimposed and mapped to the communication effectiveness requirements between different entities.

For example, sales/marketing need information from the customer to identify the CSVs (Task 1). Also, they need information from the customer about the continuous improvement effort (task 4). So Tasks 1 and 4 are mapped to the “customer and sales/marketing” interface, as the information flow between the interface necessitates supporting these tasks. Similarly, the information flow between the top management and the operations department supports tasks 2, 3 and 4 (Camp, 1993). This example shows one possible scenario. An organization needs to accurately map these tasks to the interfaces to select a suitable IS component technology.

Design of IS Component Technology

The major components of (Darnton and Giacoletto, 1992) an organizational information system include hardware, software, communication networks, policies and procedure. The design objective is to support the tasks and at the same time achieve the communication effectiveness requirements presented in Table 3. The criteria for selection of IS component technology is appropriateness, simplicity, responsiveness and, integration. Sometimes dif-

ferent alternative component technologies may be in conflict with one or more of the criteria. In those situations, a practical trade-off should be made. For example, for a discount brokerage firm the interface between the customers and the sales can be supported by regular phone calls, on-line computerized order processing using a touch-tone system, or on-line computerized order processing using PCs. All of these are appropriate IT. A regular phone system can be less responsive than the others as the brokers may be busy if there are too many phone calls. A PC based on-line system may not be simple to use and may not be widely available, but may be good in terms of responsiveness and integration. An on-line touch-tone phone is simple, responsive but may rank less in terms of integration compared to PCs. So by looking at practical concerns, the firm may decide to implement a touch-tone phone based on an on-line system.

Emerging Technologies and Solution

In the last few years, we have seen a tremendous growth and advancement in information technologies which will make communication much more effective, integrated and easier. Some of these technologies are discussed here briefly:

Internet, intranet, and extranet: The Internet is a collection of computers located all over the world that any one can access (Rosen, 1997). The Internet has been around for well over ten years (Bernard, 1997). However, the World Wide Web (www) is a recent phenomenon. With the coming of Web browsers, multimedia, graphics, push technology, and the lifting of restrictions by the government, Internet is becoming universally accepted as the communication and collaboration medium. In the near future, it may be as ubiquitous as a phone.

An Internet is open to the world, intranets are a closed network and extranets are a hybrid. (Rosen, 1997). Internet, intranets and extranets use the same technology. The difference is that, intranets only let people within the organization access their computers and extranets will allow a selected group of people outside the organization to access their network. For example, using the Internet, a person can access the home page of a business and get public information. However, by using a user name and password a customer can get pricing information through an extranet..

Data warehousing, data mining and data warehousing (Mattison, 1996) are the latest advancement in database technology. A data warehouse is a database in two senses— technical and business. One of the distinguishing features of a data warehouse is that, at the heart of data warehouse, there is a clearly defined physical database (technical understanding), which holds

within itself all information of interest to specific groups of business users (business understanding).

Data mining involves extracting the relevant information in the necessary format in a user friendly manner by systems like an executive information system (EIS), a decision support system (DSS), or an on-line analytical processing (OLAP). Another requirement for data warehousing is to be able to import and export information to other systems, which may be new and upcoming. Data warehousing technology facilitates data mining. With the proliferation of intranet and internet technology it is only logical that data mining applications will be developed based on web browsing technology.

IS/IT Design Procedure

One of the important inputs to the selection of IT is Table 3. An analysis of the tasks and communication effectiveness requirements in the table should generate a list of available IT that satisfies the requirements for the interfaces. Table 4 is an inventory of IS component technology for different communication interfaces. For example, to satisfy the communication effectiveness and task requirements for the “customer and sales/marketing interface” a toll-free number, face-to-face meeting, survey, extranet and or fax may be used. Similarly, to satisfy the requirements for the “operations to operations (different departments within operations such as, production, quality control,

Table 4: IS/IT Component Inventory

Internal/ External Entities	Customer	Sales/ Marketing	Top Mgmt.	Operations	Accounting/ Finance
Customer	User group meetings. Web-based message boards	Meetings; fax; Toll free phone Extranet survey	Meeting; Co..homepage on the Web; Internet	Meetings; Data mining; Data warehousing; Intranet	Extranet; Data mining; EDI
Sales/Marketing		Meetings; Fax: Intranet	Meetings; Intranet	Meetings; Intranet; Data mining;	Meetings; Data mining; Intranet
Top Management			Meetings; Intranet; ESS	Meetings; Intranet	Meetings; Data mining; Intranet
Operations				Meetings; Data mining; Data warehousing; DSS; Intranet	Meetings; Data mining; Intranet
Accounting/ Finance					Meetings; Data warehousing; Data mining; Intranet; DSS

design etc.)” the appropriate techniques and technologies may include face-to-face meetings, decision support system (DSS), intranet and data mining and data-warehousing. Once a complete inventory of IS/IT component technologies for the communication interfaces is made, it is important to evaluate these based on the criteria established.

INTEGRATION

Organizational integration (Carlson, 1979; Cooper and Zmud, 1990) of IS/IT for TQM implementation should have the following three major focuses: 1) technological integration—integration of IT components (Bullers, 1991) like data, applications, hardware, software and communications network; 2) functional integration—coordination of operations of different entities of the firm; and 3) strategic integration—alignment of firm’s strategy (Sabbaghi, 1991) with the operations of functional units.

Technological integration is becoming more possible as database and networking technologies are becoming increasingly advanced and affordable. From Table 4, it can be seen that many of the communication interfaces utilize Web-based IT like internet, intranet, and extranet. Recently, there has been tremendous progress in database technology leading to decision supporting tools like data mining and data warehousing. These will facilitate technological integration. One key consideration to achieve technological integration is standardization and compatibility between different IS/IT components such as hardware, software and networks.

Functional integration can be achieved by a certain degree of technological integration supplemented by empowered policies and procedures. For example, advanced database technology like data mining and data warehousing is needed to integrate the information flow between sales/marketing, top management, operations, accounting/finance. For effective functional integration, proper policies and procedures have to be instituted for access and update of the database, reporting and empowered decisionmaking by the managers in different functional areas.

Strategic integration involves the orientation of all entities of the firm to achieve the TQM objectives. Strategic integration through IS should be geared toward effective communication and dissemination of top management’s priorities and goals to the functional department. If an organization has achieved a certain degree of technological and functional integration, then strategic integration is more dependent on transforming organizational culture toward empowered decisionmaking through effective communication.

An Example of Integration through IS

An example is used to illustrate how integration can take place in a factory setting. The example is biased towards manufacturing as the manufacturing sector has embraced the TQM philosophy to a large extent (Carlson, 1979). However, the concepts are also applicable in the service sector. Figure 3 illustrates IS integration at different managerial levels. The factory floor data collection system collects operational information from different cells on a continuous basis. Two examples of operational information are (a) defect percentage in the output, and (b) types of defects detected on the assembly line. This information is kept in a relational database. There are about 5,000 instruments and sensors distributed throughout the floor and about a million data records are collected every day. The engineering department keeps its product specifications in an object-oriented database. This includes information such as CAD designs, CAM specifications and production plans. The database tracks Design- for-Manufacturing characteristics and cost figures for parts that fall under category A of A-B-C analysis. The engineering department revises design specifications for improved customer satisfaction and production efficiency. The CAD/CAM database stores information about 250,000 parts and about 3,000 parts are revised every day. Of these changes, 500 are significant changes while the rest are minor changes.

The marketing department relies on two databases. The first one is a relational database, which tracks more than 100,000 orders every day. This sales and distribution database maintains customer information, daily ongoing pre-sales, sales and post-sales activities, and accounts receivables and credit management details. About 5 million updates, inserts and deletes are generated every day. The second database is a textual database, which contains product proposals, product changes, customer feedback, questions, and issues.

In order to effectively manage high data volumes, a data warehouse is used. This data warehouse is the underlying engine for an executive decision support system. Operations managers review daily operational statistics related to quality control. Engineering managers analyze the most significant design changes in light of stated goals at the individual cell, production unit, and factory level. Marketing managers keep a tab on the short-term and long-term trends in pre-sales, sales, and post-sales activities. Relevant information from these databases is available to IS users through a corporate intranet. Compliance information for ISO 9000 and other relevant certifications is managed from a centralized intranet browser front end.

Figure 3. Integration of IS for TQM

Customers have multiple options to communicate with the corporation, namely, voice, fax, e-mail and web. First, they can converse with the employees in person and through phone. Employees enter such interactions directly into the database. Second customers can send faxes to employee's personal fax number. Fax transmissions are electronically received and stored immediately into the textual database as images. Using optical character

recognition (OCR) technology, textual information in these images is retrieved for indexing purposes. Each employee reviews his or her incoming fax transmission and acts on it. Using workflow technologies, they forward the necessary documents to the appropriate persons on the workflow routings. Third, customers can e-mail to employees through internet connections. Finally, they have the option of directly placing orders and posting issues on the corporate web site. Firewalls and encryption technologies minimize unauthorized changes and protect the quality of communications.

A customer is typically assigned to one single employee who oversees all the necessary internal work processes. This minimizes the chances of a customer being passed from one employee to another. Employees are not mired in the process of maintaining the information such as storing documents in physical file cabinets. They have easy access to required information on the corporate intranet. Managing information in this way ensures that customers receive a high degree of satisfaction from their communications and interactions with the firm.

Employees are empowered to perform necessary activities to ensure customer satisfaction. For example, if a customer wants a functional enhancement of a product, an employee is authorized to approve changes that will cost up to 50% of the product contributions from sales to that customer in the past six months (subject to a ceiling of \$100,000). The information system provides access to historical sales and contribution information and design change costs. Over a period of time, managers periodically review different workflow processes based on historical information and set up empowering policies and procedures for employees. A distributed and heterogeneous information system environment thus supports various TQM activities from design to delivery.

CONCLUSION

TQM is a strategic philosophy whose main premise is to achieve competitiveness through customer satisfaction. To compete, an organization should embark on a proactive strategy where every entity is responsible for continuous improvement. This requires an effective communication system that will make the organization responsive to customer expectations and needs.

The framework described in this paper elaborates the steps in selecting IT components and designing an effective IS for TQM. This may be useful to the managers for accelerating the process of TQM implementation.

We have outlined an approach to effective communication integration for TQM through the use of IS. Future research can identify specific combinations of IS/IT components that will be useful in different organizational situations. Some questions of interest that need empirical research are : (a) What are the benefits of integration through IS/IT in TQM implementation? (b) Which technologies are most useful for promoting integration in TQM implementation? (c) What organizational factors promote or inhibit certain combination of IS/IT. A contingent approach for developing an IS for TQM implementation may evolve from studying use of IS/IT in different organizations.

REFERENCES

- Adam, Jr., E. E. (1994). Alternative quality improvement practices and organization performance. *Journal of Operations Management*, 12, 27-44.
- Ali, R., Montazemi, G., & Miltenburg, J. (1991). A Modeling Tool for Analyzing the Information Requirements of Computer Integrated Manufacturing Systems. *Infor*, 29, 240-250
- Ayres, J. B. (1993). TQM and Information Technology: Partners for Profit. *Information Strategy*, 9, 26-31.
- Bernard, R. (1997). *The Corporate Intranet*. John Wiley & Sons, New York.
- Blest, J. P., Hunt, R. G. and Shadle, C. C. (Spring, 1992). Action Teams in the Total Quality Process: Experience in a Job Shop. *National Productivity Review*, 195-202.
- Bullers, W., Jr. (1991). A Tripartite Approach to Information Systems Development. *Decision Science*, 22, 120-135.
- Camp, R. C. (1989). Competitive Bench Marking: Xerox's Powerful Quality Tool is Making Total Quality Happen, Research Report, *The Conference Board*, 35-42.
- Camp, R. C. (1993). *Benchmarking: The Search for Industry Best Practices that Lead to Superior Performance*. Milwaukee, Wisconsin: ASQC Quality Press.
- Carlson, W. M. (Spring, 1979). Business Information Analysis and Integration Technique (BAIT) -The New Horizon. *Database*, 3-10.
- Caudron, S. (February, 1993). Keys to Starting a TQM Program. *Personnel Journal*, 28-35.
- Chalk, M. B. (1993). Implementing Total Quality Within Corporate Real Estate. *Site selection and Industrial development*, 38, 433-436.

- Chung, R. K. (May, 1989). Benchmarking Performance in the Credit Function. *Business Credit*, 19-21.
- Conger, J.A., and Kanungo, R. N. (1988). The Empowerment Process: Integrating Theory and Practice. *Academy of Management Review*, 13(3), 471-482.
- Cooper, R. B., and Zmud, R. W. (1990). Information Technology Implementation Research: A Technological Diffusion Approach. *Management Science*, 36, 123-139.
- Darnton, G. and Giacoletto, S. (1992). *Information in the Enterprise*. Boston, Digital Press.
- Dertouzos, M. I., Lester, R. K. and Slow, R. M., and the MIT Commission on Industrial Productivity. (1989). *Made in America*. Cambridge, MA: MIT Press.
- Doll, W. J. & Vonderembske, M.A. (June, 1987). Forging a Partnership to Achieve Competitive Advantage: The CIM Challenge. *MIS Quarterly*, 205-220.
- Evans. J.R., and Lindsay, W. L. (1996). *The Management and Control of Quality*, (3rd ed.). St. Paul, Minnesota: West Publishing Company.
- Flynn, B. B. (1994). A framework for quality management research and an associated measurement instrument. *Journal of Operations Management*, 11, 339-366.
- Frey, R. (September, 1993). Empowerment or Else. *Harvard Business Review*, 80-89.
- Garvin, D. A. (November, 1987). Competing on the eight dimensions of quality. *Harvard Business Review*, 101-109.
- Goldhar, J. D. and Jelnik, M. (1985). Computer Integrated Flexible Manufacturing: Organizational, Economic, and Strategic Implications. *Interfaces*, 15, 94-105.
- Hart, C. and Schlesinger, L. (1991), Total Quality Management and the Human Resource Professional: Applying the Baldrige Framework to Human Resources. *Human Resource Management*, 30, 433-454.
- Hendricks, C. F., and Triplett, A. (December, 1989). TQM: Strategy for 90s Management. *Personnel Administrator*, 42-48.
- Huston, L. A. (September, 1992). Using Total Quality to Put Strategic Intent into Motion. *Planning Review*, 21-23.
- Kettinger, W. J., Grover, V., Guha, S., and Segars, A. H. (March 1994). Strategic Information Systems Revisited: A Study in Sustainability and Performance. *MIS Quarterly*, 31-58.
- King, W. R., and Thompson, S.H. T. (1994). Facilitators and inhibitors for the strategic use of information technology. *Information and Management*, 27, 71-87.

- Kivajarvi, H., and Saarinen, T. (1995). Investment in Information System and the Financial Performance of the Firm. *Information and Management*, 28, 143-163.
- Leddick, S. (Winter, 1991). Teaching Managers to Support Quality-Improvement Efforts. *National Productivity Review*, 69-74.
- Levine, M. J. (February, 1992), .Labor and Management Response to Total Quality Management. *Labor Law Journal*, 107-116.
- Ludeman, K. (December, 1992). Using Employee Surveys to Revitalize TQM. *Training*, 51-57.
- Maccoby, M. (May, 1992). Creating an Empowered Organization. *Research, Technology Management*, 50-51.
- Marquardt, I. A. (August, 1992). Inside The Baldrige Award Guidelines. *Quality Progress*, 93-96.
- Mathers, H. (August, 1991). Don't Just Satisfy, Delight Your Customers. *APICS - The Performance Advantage*, 22-25.
- Mattisn, R. (1996). *Data Warehousing: Strategies, Technologies, and Techniques*, New York: McGraw Hill.
- Meredith, J. R. (1992). *The Management of Operations: A Conceptual Emphasis*, New York: Wiley Inc.
- Patten, T. H. (Winter, 1991). Beyond Systems - The Politics of Managing in a TQM Environment. *National Productivity Review*, 9-19.
- Powell, T. C. (1995). Total Quality Management as a Competitive Advantage: A Review and Empirical Study. *Strategic Management Journal*, 16, 15-37.
- Rooney, C. (1990). Successfully implementing TQM. *American Paint and Coating Journal*, 74, 36-40.
- Rosen, A. (1997). *Looking into Intranets & the Internet*, American Management Association, New York.
- Ross, J. E. (1993). *Total Quality Management: Text, Cases and Readings*, Delray Beach, Florida; St. Lucie Press.
- Sabbaghi, A. (1990). CIM Strategy and Strategic Management: An MIS Perspective. *Journal of Applied Business Research*, 1, 57-66.
- Sayeed L. and Brightman, H. J. (1994). Can Information Technology Improve Managerial Problem Finding. *Information and Management*, 27, 377-390.
- Schonberger, R.J. (1983). Work Improvement Programs: Quality Control Circles compared with traditional Western approaches. *International Journal of Operations and Production Management*, 3, 18-32.
- Schonberger, R. J. (1992). Is strategy strategic? impact of total quality management on strategy. *Academy of Management Executive*, 6, 80-87.

- Tillery, K.R. (1985). *An Exploratory Study of the Quality Function with Five Manufacturing Organizations*, Unpublished doctoral dissertation, Georgia State University.
- Tobin, L. M. (November, 1990). The New Quality Landscape, Total Quality Management. *Journal of Systems Management*, 10-14.
- Vansina, L. S. (Winter, 1990). Total Quality Control: An Overall Organizational Improvement Strategy. *National Productivity Review*, 59-74.

Chapter 11

Methodology of Schema Integration for New Database Applications: A Practitioner's Approach

Joseph Fong
City University of Hong Kong

Kamalakar Karlapalem
Hong Kong University of Science & Technology

Qing Li and Irene Kwan
Hong Kong Polytechnic University

A practitioner's approach to integrate databases and evolve them so as to support new database applications is presented. The approach consists of a joint bottom-up and top-down methodology; the bottom-up approach is taken to integrate existing database using standard schema integration techniques (B-Schema), the top-down approach is used to develop a database schema for the new applications (T-Schema). The T-Schema uses a joint functional-data analysis. The B-schema is evolved by comparing it with the generated T-schema. This facilitates an evolutionary approach to integrate existing databases to support new applications as and when needed. The mutual completeness check of the T-Schema against B-Schema derive the schema modification steps to be performed on B-Schema to meet the requirements of

the new database applications. A case study is presented to illustrate the methodology.

There has been a proliferation of databases in most organizations. These databases are created and managed by the various units of the organization for their own localized applications. Thus the global view of all the data that is being stored and managed by the organization is missing. Schema integration is a technique to present such a global view of an organization's databases. There has been a lot of work done on schema integration. Batini et al. (1986) and Özsu and Valduriez (1991) present surveys of work in this area. But all these techniques concentrate on integrating database schemas without taking into consideration of new database applications. This paper presents a practical approach to schema integration to support new database applications by comparing the existing databases against data requirements of the new applications. If the existing databases are inadequate to support new applications, then they are evolved to support them.

In any schema integration methodology all the database schemas have to be specified using the same data model. The proposed approach uses an extended entity relationship (EER) data model. Therefore, the first step in the schema integration methodology is to translate a non-EER database schema to an EER database schema. A joint bottom-up and top-down approach for schema integration to support new database applications is proposed. The bottom-up approach is taken to integrate existing databases using standard schema integration techniques whereas the top-down approach is used to come up with the database schema for the new applications. The top-down approach uses the joint functional-data analysis. The B-schema generated by bottom-up approach is evolved by comparing it with the T-schema generated by the top-down approach. This facilitates a stream lined approach to evolve integrated databases to support new applications.

Conventional approaches that have been widely used in database community for database design can be classified as top-down, and are therefore suitable for designing databases from scratch to support new applications. On the other hand, research in the area of heterogeneous distributed databases over the last decade has emphasized on bottom-up approaches towards global schema derivation by integrating existing databases. These two kinds of approaches are complementary in many aspects, and thus can be combined into a unified framework for schema integration.

Fong et al. (1994) developed a hierarchical comparison scheme using three major criteria for comparing relationships in two schemas. The paper classified the relationship integration by taking into account the degree of

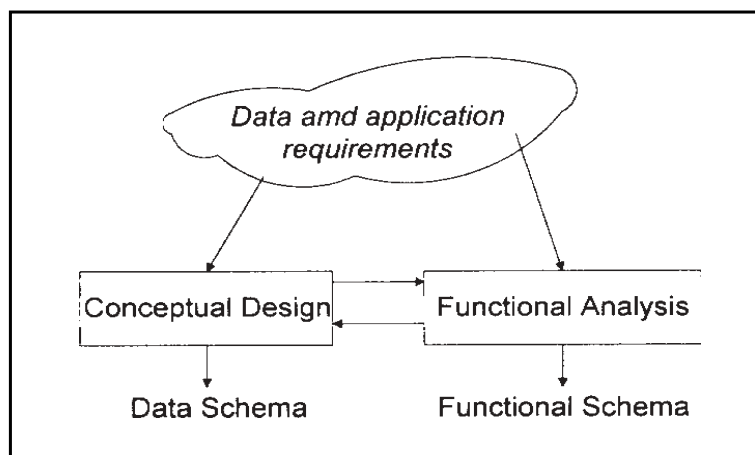
relationship, roles and structural constraints as the main features to guide the comparison of relationships. Fong (1992) applied information capacity equivalence as a measure of correctness for judging transformed schemas in schema integration. It presents a classification of common integration based on their operational goals and derive from them the instances level of equivalence of schemas after integration.

Top-down schema design techniques

Traditional database design has focused on data elements and their properties, and the approaches taken by database professionals were data-driven; the entire focus of the design process is placed on data and their properties (Korth and Silberschatz, 1991; Ullman, 1982; Elmars and Navathe, 1989). Typically, a data-driven (DD) approach first creates a conceptual schema by analyzing data requirements, which is then followed by logical and physical schema design; the applications that use the database will be developed after the database is created. An alternative kind of design that has been very popular in information systems design is termed as function-driven (Senn, 1989). In these kind of approaches, the main focus is on applications rather than data. More specifically, functional analysis starts with application requirements to generate functional schemas, which are then mapped into application specifications. These form the basis for the subsequent application program design. In functional analysis, databases are seen as isolated repositories of information used by individual activities; the vision of data as a global resource of the enterprise is not present.

More recently, the idea of applying functional analysis techniques and concepts from traditional information systems area into conceptual database design has become increasingly popular, and has resulted in so-called Joint Data- and Function-Driven (JDfD) approach, which is more powerful than

Figure 1: Joint data and function driven approach to database design



pure DD approach (Batini et al., 1986). As shown in Figure 1, JDfD produces the conceptual database structure and the function schema in parallel, so that the two design processes influence each other. More specially, the JDfD approach makes it possible to test whether data and function schemas are mutually consistent and complete. Note that both pure DD and JDfD types of approaches are used for designing new databases to support new applications.

Bottom-up schema integration techniques

Schema integration is a relatively recent problem that has appeared in the context of distributed, heterogeneous databases (Sheth and Larson, 1990; McLoed and Heimbigner, 1980). It takes place in a bottom-up fashion, requiring that an integrated global schema be designed from the local schemas, which refer to existing databases. Figure 2 summarizes the schema integration activity which has as input the local schemas and local transactions, and has as its output the global schema as well as the specifications of data and query-mapping from global to local databases.

Though different researchers have proposed different solution procedures for conducting schema integration, they can be eventually considered to involve a mixture of the following activities: pre-integration, comparison, conformation, and integration (Batini et al., 1986). Hence schema integration in general involves such four steps. Note that the emphasis of such a bottom-up approach towards generating a global schema has been on identifying the correspondences and inter-schema properties, and that the subsequent integration of the local databases schemas is aimed to satisfy the criteria of completeness and correctness, minimality, and understandability. The bottom-up approach is to sum up the capacity of existing databases in one global schema.

Figure 2: Bottom-up schema integration methodology

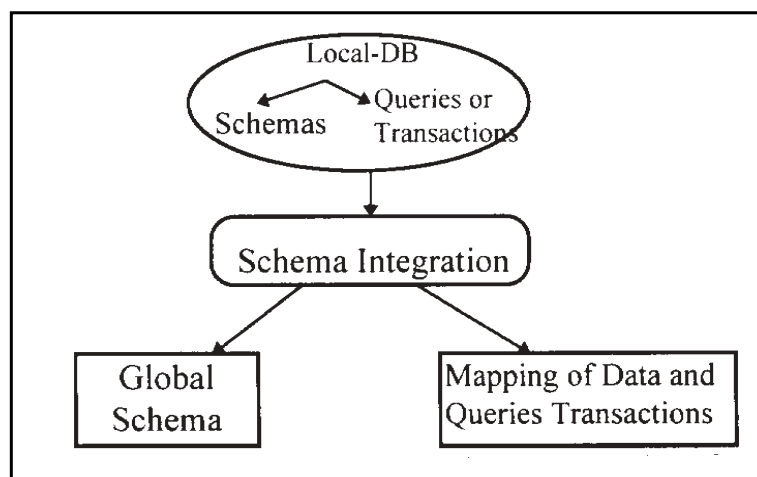
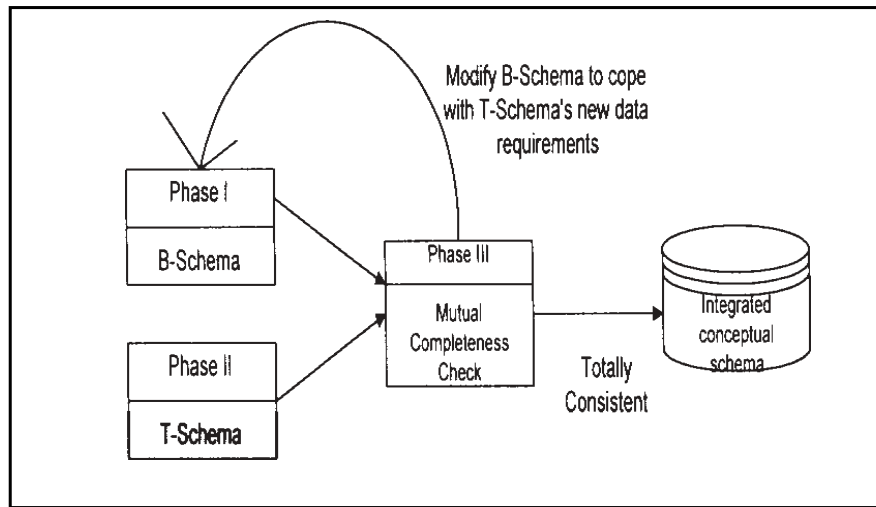


Figure 3: Context diagram for schema integration methodology

Our approach

The above overviews of the two different kinds of approaches for database schema design and integration demonstrate that they have very different emphasis and focuses, reflecting their different problem domains. Yet they are complementary from a database integration point of view. Indeed it is the theme of this paper to combine these two sorts of approaches into a unified framework for the database integration purpose. The purpose is to develop a practical methodology to integrate, and to be able to evolve existing databases by considering the new requirements from the applications to be developed and supported. More specifically, the purposed mutual completeness checking methodology can be described as in the Figure 3 (Ozkarahan, 1990).

Phase I. Integrate existing database schemas using a bottom-up schema integration methodology; the resultant global schema is called B-schema;

Phase II. Design a conceptual schema using the Joint Data- and Function-Driven methodology for the applications to be developed and supported; the resultant schema is called T-schema;

Phase III. Evaluate the B-schema by comparing it with the T-schema obtained in phase II, checking both consistency and completeness of B-schema with respect to the T-schema. If the two schemas are totally consistent, then the B-schema can be the integrated schema. Otherwise refine the B-schema by resolving any conflicts and incorporating any new concepts that are in T-schema but not in B-schema.

PHASE I - B-SCHEMA DESIGN

Algorithm:

Begin

For each existing database do /* step 1 */

If its conceptual schema does not exist

then reconstruct its conceptual schema in EER model
by reverse engineer;

For each pair of existing databases' EER models of
schema A and schema B do

begin

Resolve the semantics conflicts between the schema A and schema B;

/* step 2 */

Merge the entities between the schema A and schema B; /* step 3 */

Merge the relationships between the schema A and
schema B; /* step 4 */

end

In the first step, the conceptual schema of existing databases may or may not exist, and may also be out-of-date. Thus, the first step of schema integration is to reconstruct the conceptual schema, using the EER models from the logical schema of the various data models (such as hierarchical, network or relational). The second step is to merge them into one EER model and preserve all the semantics of the existing databases. This can be done iterative by merging each pair of EER models into one. However, the conflicts among the EER models must be resolved before merging. This process requires users intervention to solve the contradictions in the semantics. The third step is to associate the entities of different EER models by relationship. By doing so, the relationship of EER models can be captured in one EER model along with the semantics. The fourth step is to associate the relationships among EER models by absorbing one into another or by their related semantics such as subtype, generalization, aggregation and categorization.

Step 1. Reverse engineer logical schema to conceptual schema.

Input: the DDL(data description language) statements of the existing hierarchical, network or relational database to the reverse engineer

Output: the conceptual schema in EER model

Logical schema describes the data structure of databases. They only represent few semantics in the databases. Conceptual schema in EER model carries more semantics than logical schema. To rebuild an EER model from a logical schema is a reverse engineering process. User involvement is needed

to confirm the implied semantics in the logical schema. Basically, the constraints and semantics in the logical schema will be preserved in the reconstructed EER model. Refer to Fong (1992) and Batini et al. (1992) for detailed methodology in the process.

Step 2. Resolve conflicts among EER models.

Input: Schema A and B with entities Es and attributes As in conflicts to each other on semantics

Output: Integrated schema X after data transformation

Step 2.1 Resolve conflicts on synonyms and homonyms

For each pair of (E_a, E_b) Do

For each $x \in (\text{Attr}(E_a) \cap \text{Attr}(E_b))$, Do

IF $E_a.x$ and $E_b.x$ have different data types or sizes

THEN x in E_a and E_b may be homonyms, let users clarify x in E_a and E_b ;

For each pair of (E_a, E_b) , Do

For each pair (x, y) , $x \in \text{Attr}(E_a)$ and $y \in \text{Attr}(E_b)$, Do

IF $x \neq y$, and $E_a.x$ and $E_b.y$ have the same data type and size

THEN $((x, y)$ may be synonyms, let users clarify (x, y));

In some cases, we can consider the derived data as synonyms by matching their domain value based on conditions. For instance, two attributes A_a and A_b can be taken as synonyms by applying the following constraint rules in a stored procedure:

IF $A_a > 74$ mark /* student mark above 74 is a “A” grade */

THEN $A_b = \text{‘A’ grade}$

IF $75 \text{ mark} > A_a > 64 \text{ mark}$ /* student mark above 64 is a “B” grade */

THEN $A_b = \text{‘B’ grade}$

Step 2.2 Resolve conflicts on data types

Case 1: Conflict: an attribute appears as an entity in another schema

Resolution:

For each pair of (E_a, E_b) , Do

IF $x \in (A.A_b \cap B.E_b)$ /* attribute A_b in schema A appears as entity E_b in schema B */

THEN cardinality $(E_a, E_b) \sim n:1$;

Case 2: Conflict: a key appears as an entity in another schema

Resolution:

For each pair of (E_a, E_b) , Do

IF $x \in (\text{keys}(A.A_b) \cap B.E_b) /* \text{entity key } A_b \text{ in schema A}$
 appears as entity E_b in schema B */

THEN $\text{cardinality}(E_a, E_b) \leftarrow 1:1;$

Case 3: Conflict: a component key appears as an entity in another schema

Resolution:

For each pair of (E_a, E_b) , Do

IF $(x \subset \text{keys}(A.A_b)) \cap B.E_b /* \text{entity key component } A_b \text{ in}$
 schema A appears as entity E_b in schema B */

THEN $\text{cardinality}(E_a, E_b) \leftarrow m:n;$

Step 2.3 Resolve conflicts on key

Conflict: A key appears as a candidate key in another schema

Resolution:

For each pair of (E_a, E_b) , Do

IF $x \in (\text{key}(A.A_c) \cap B.A_c)$

THEN Let users clarify x in E_a and E_b ;

Step 2.4 Resolve conflicts on cardinality

Conflict: Identical entities with different cardinality in two schemas

Resolution:

For each pair of $\text{cardinality}(A.E_a, A.E_b)$ and $(B.E_a, B.E_b)$, Do

IF $(\text{cardinality}(A.E_a, A.E_b) = 1:1) \wedge (\text{cardinality}(B.E_a,$
 $B.E_b) = 1:n)$

THEN $\text{cardinality}(X.E_a, X.E_b) \leftarrow 1:n;$

For each pair of $\text{cardinality}(A.E_a, A.E_b)$ and $(B.E_a, B.E_b)$, Do

IF $(\text{cardinality}(A.E_a, A.E_b) = 1:1 \text{ or } 1:n) \wedge$
 $(\text{cardinality}(B.E_a, B.E_b) = m:n)$

THEN $\text{cardinality}(X.E_{a1}, X.E_{a2}) \leftarrow m:n;$

Step 2.5 Resolve conflicts on weak entities

Conflict: A strong entity appears as a weak entity in another schema

Resolution:

For each pair of (E_a, E_b) , and their relationships: R in schema A and B,

Do IF $((x \in \text{key}(A.E_a)) \cap ((x \notin \text{key}(E_{a2}))) \wedge$

$((x \in \text{key}(B.E_a)) \cap ((x \subset \text{key}(B.E_b)))$

 THEN $\exists (x \in \text{key}(E_a)) \cap ((x \subset \text{key}(E_b)))$

In other words, the key of the weak entity must concatenate with its strong entity key.

Step 2.6 Resolve conflicts on subtype entities

Conflict: A subtype entity appears as a supertype entity in another schema

Resolution:

For each pair of (E_a, E_b) Do

IF $((\text{domain}(A.E_a) \subseteq \text{domain}(A.E_b)) \wedge ((\text{domain}(B.E_b) \subseteq \text{domain}(B.E_a)))$

THEN $\text{cardinality}(X.E_a, X.E_b) \leftarrow 1:1 /* A.E_a = E_a \text{ in A schema } */$

The above step 2.1 to 2.6 can be illustrated in Figure 4.

Step 3 Merge entities

Input: existing entities in schema A and B

Output: merged (connected) entities in (integrated) schema X

Step 3.1 Merge entities by union

Condition: Identical entities with different attributes in two schemas

Integration:

For each pair of (E_a, E_b) , Do

IF $((\text{domain}(A.E_a) \cap \text{domain}(B.E_b)) \neq 0)$

THEN $\text{domain}(X.E_a) \leftarrow \text{domain}(A.E_a) \cup \text{domain}(B.E_b);$

Step 3.2 Merge entities by generalization

Case 1: Condition: Entities with same attributes appear in two schemas, but instance of the first entity in one schema cannot appear in another schema (i.e. disjoint generalization)

Integration:

For each pair of (E_a, E_b) , Do

IF $((\text{domain}(E_a) \cap \text{domain}(E_b)) \neq 0) \wedge ((x \in \text{instance}(E_a))$

$\wedge (x \notin \text{instance}(E_b)) \wedge ((y \in \text{instance}(E_b)) \wedge$

$(y \notin \text{instance}(E_a)))$

THEN begin $\text{domain}(E_x) \leftarrow \text{domain}(E_a) \cap \text{domain}(E_b)$

$((x \in \text{instance}(X.E_a) \wedge (x \notin \text{instance}(X.E_b))$

$((y \in \text{instance}(X.E_b)) \wedge (y \notin \text{instance}(X.E_a)));$

end

Case 2: Condition: Entities with same attributes appear in two schemas, but instance of the first entity in one schema can appear in another schema (i.e.

Figure 4: Examples on conflicts resolutions by data transformation

Step	Before data transformation	After data transformation
2.1 EER model with synonyms and homonyms	<p>Schema A Schema B</p>	<p>Schema X <small>Synonyms: E_a, E_b Homonyms: A_c, A_d</small></p>
2.2 EER models in data type conflicts	<p>Case 1</p> <p>Schema A Schema B</p> <p>Case 2</p> <p>Schema A Schema B</p> <p>Case 3</p> <p>Schema A Schema B</p>	<p>Case 1 Schema X</p> <p>Case 2 Schema X</p> <p>Case 3 Schema X</p>
2.3 EER models in key conflicts resolved with users confirmation	<p>Schema A Schema B</p>	<p>Schema X</p>
2.4 EER models in cardinality conflict	<p>Schema A</p> <p>Schema B</p>	<p>Schema X</p>
2.5 EER models in weak entity conflict	<p>Schema A</p> <p>Schema B</p>	<p>Schema X</p>
2.6 EER models in subtype conflicts (where > means isa relationship)	<p>Schema A</p> <p>Schema B</p>	<p>Schema X</p>

Figure 5: Examples on entities merge into an integrated schema

Step	Before entities merge	After entities merge
3.1 Merge EER models by union	<p>Schema A Schema B</p>	<p>Schema X</p>
3.2 Merge EER models by generalization	<p>Schema A Schema B</p>	<p>Schema X</p>
3.3 Merge EER models by subtype	<p>Schema A Schema B</p>	<p>Schema X</p>
3.4 Merge EER models by aggregation	<p>Schema A MVD: $R_d \twoheadrightarrow E_a$</p>	<p>Schema X</p>
3.5 Merge EER models by categorization	<p>Schema A Schema B</p>	<p>Schema X</p>
3.6 Merge EER models by implied relationship	<p>Case 1 Schema A Schema B</p> <p>Case 2 Schema B</p>	<p>Case 1 Schema X</p> <p>Case 2 Schema X</p>

overlap generalization)

Integration:

For each pair of (E_a, E_b) , Do

IF $((\text{domain}(E_a) \cap \text{domain}(E_b)) \neq 0)$

THEN $\text{domain}(E_x) \leftarrow \text{domain}(E_a) \cap \text{domain}(E_b)$;

Step 3.3 Merge entities by subtype relationship

Condition: Two subtype related entities appear in two different schemas

Integration:

For each pair of (E_a, E_b) , Do

IF $\text{domain}(E_a) \subseteq \text{domain}(E_b)$

THEN $E_a \text{ isa } E_b$; /* entity E_a is a subset of entity E_b */

Step 3.4 Merge entities by aggregation

Condition: A relationship in one schema is related to an entity in another schema

Integration:

For each pair of entity A and relationship B, Do

IF $(\exists \text{MVD: } R_b \twoheadrightarrow E_b)$ /* MVD means multivalued dependency (Fong, 1992) */

THEN begin $E_x \leftarrow (E_{b1}, R_b, E_{b2})$ /* aggregation */
 $\text{cardinality}(E_x, E_a) \leftarrow 1:n$;

end

Step 3.5 Merge entities by categorization

Condition: One entity in one schema is subtype related to two entities in another schema

Integration:

For each group of (E_{a1}, E_{a2}, E_b) , Do

IF $(\text{instance}(E_b) \subseteq \text{instance}(E_{a1})) \wedge$
 $(\text{instance}(E_b) \subseteq \text{instance}(E_{a2}))$

THEN begin $E_c \leftarrow (E_{a1}, E_{a2})$ /* categorization */

$(\text{instance}(E_b) \text{ isa } \text{instance}(E_{x1})) \wedge$

$(\text{instance}(E_b) \text{ isa } \text{instance}(E_{x2}))$; /* E_b is subset to
 E_{a1} or E_{a2} */

end

Step 3.6 Merge entities by implied binary relationship

Condition: An identical data item appears in different data types in two schemas

Integration:

For each pair of (E_a, E_b) , Do

IF $x \in (A.A_d \cap \text{key}(B.A_d))$

THEN Cardinality $(E_a, E_b) \leftarrow n:1$;

Condition: Two identical data items appear in different data types in two schemas

Integration:

For each pair of (E_a, E_b) , Do

IF $(x \in (A.A_d \cap \text{key}(B.A_d))) \wedge (y \in (\text{key}(A.A_c) \cap B.A_c))$

THEN Cardinality $(E_a, E_b) \leftarrow 1:1$;

The above step 3.1 to 3.6 can be illustrated in Figure 5.

Step 4 Merge relationships

Input: existing relationships in schema A and B

Output: merged(connected) relationships in integrated schema X

Step 4.1 Merge relationships by subtype relationship

Case 1: Condition: Two relationship R_a, R_b are in the same role with different level of participation (Elmarsr and Navathe, 1989)

Integration:

For each pair of (R_a, R_b) , Do

IF $((\text{Cardinality}(A.E_a, A.E_b) = 1:n) \wedge (\text{Cardinality}(B.E_a, B.E_b) = 1:(0,n)))$

THEN begin $\exists E_c$ Cardinality $(E_a, E_c) \leftarrow 1:n$
 $E_c \text{ isa } E_b$

end

Case 2: Condition: Two relationships have different semantics but with intersecting relationship

Integration:

For each pair of relationships (R_a, R_b) , Do

IF $((\text{domain}(E_a) \cap \text{domain}(E_b)) \neq \emptyset)$

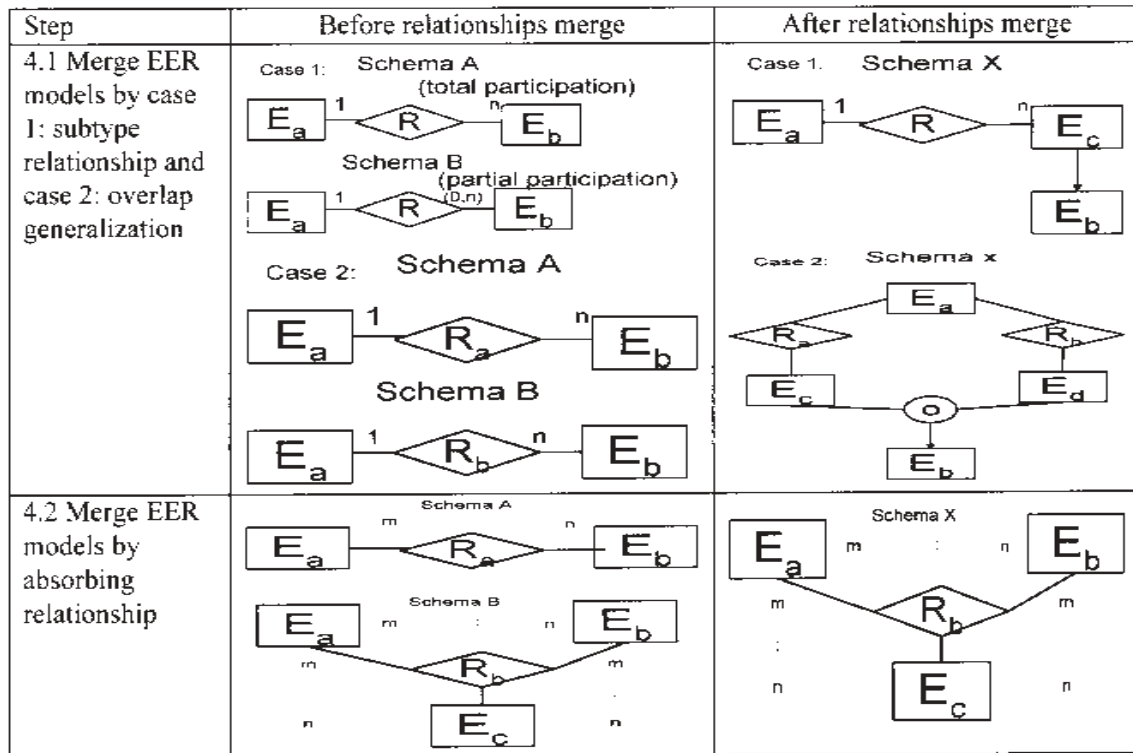
THEN begin $E_c \leftarrow R_a$

$E_b \leftarrow R_b$

$E_d \text{ isa } E_b$

$E_c \text{ isa } E_b$

Figure 6: Merge relationships for schema integration



Step 4.2 absorbing lower degree relationship into a higher degree relationship

Condition: The semantic of a lower degree relationship is implied in the semantics of a higher degree relationship

Integration:

For each pair of (R_a, R_b) , Do

IF $((\text{domain}(R_b) \subset \text{domain}(R_a)) \wedge$
 $(\text{degree}(R_b) < \text{degree}(R_a)))$

THEN $X.R_b \leftarrow R_a$

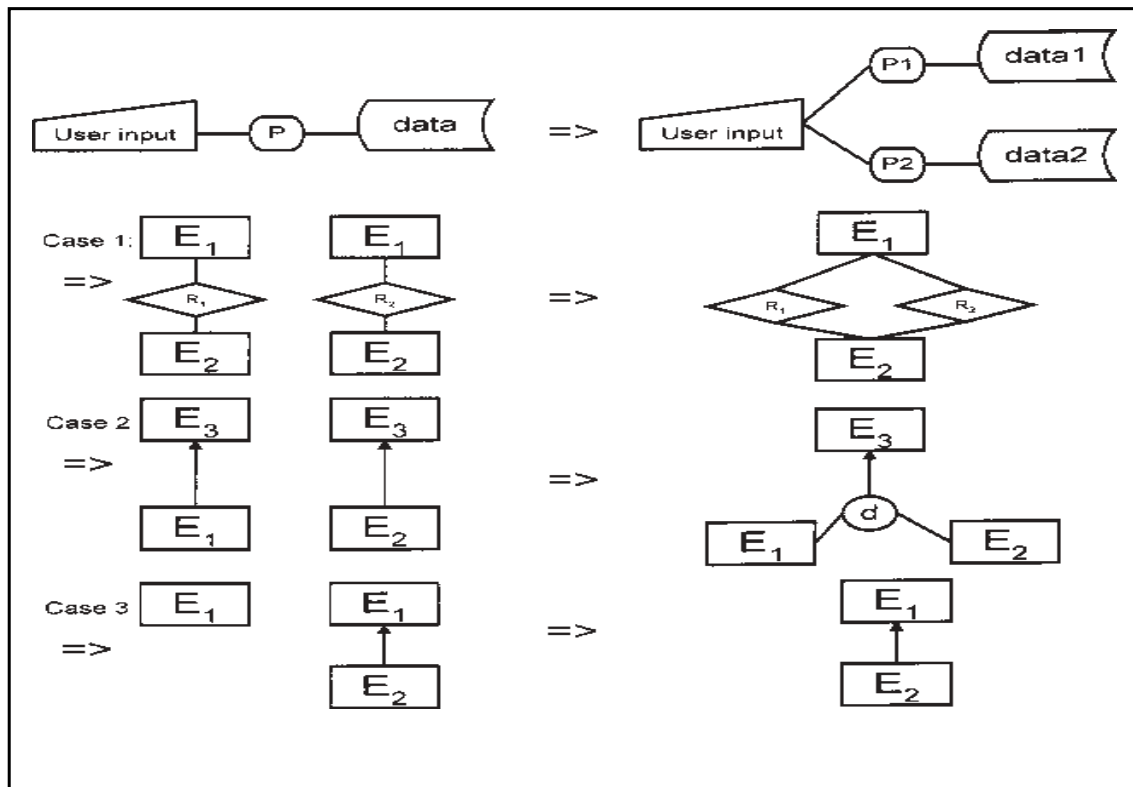
Note: in case $\text{domain}(R_b) \not\subset \text{domain}(R_a)$ provided that certain constraints are met for reason of their semantics, these constraints must be preserved in the integrated schema.

The above step 4.1 to 4.2 can be illustrated in Figure 6.

PHASE II-T-SCHEMA DESIGN

T-Schema design methodology is used to derive with a database schema for the data and process requirements of new set of applications. This methodology is based on joint functional and data analysis. It consists of

Figure 7: Refine Data Schema by process decomposition



evaluating the data and process requirements of the applications and incrementally developing the database schema by applying transformations to it. There are two approaches for this joint functional and data analysis: data driven and function driven (Ozkarahan, 1990). This methodology chooses function driven for its simplicity. Its procedure is to decompose a main process into sub-processes in a DFD, i.e. functional schema. Then it refines the data analysis for each sub-process. The sum of these data analysis is a refined data analysis, i.e. data schema, for the refined function analysis. They can be described as follows:

Step 1: Identify a main process and its data requirements for a new application
Define a main process P in a DFD and its data requirements in an EER model.

Step 2: Refine the new application's functional schema using a DFD and data schema using in the EER model.

Decompose P into sub-processes P_1, P_2, \dots, P_n in a functional schema;
For each sub-process P_i , Do
IF P_i requires $E_{i1}, E_{i2}, \dots, E_{ik}$ and $R_{i1}, R_{i2}, \dots, R_{ij}$ R_s
are relationships between entities E_s */
THEN integrate $E_{i1}, E_{i2}, \dots, E_{ik}$ and $R_{i1}, R_{i2}, \dots, R_{ij}$
into a refined EER model.

- Case 1: The data analysis of two sub-processes can be integrated by binary relationship
- Case 2: The data analysis of two sub-processes can be integrated by generalization
- Case 3: The data analysis of two sub-processes can be integrated by subtype relationship

PHASE III – MUTUAL COMPLETENESS CHECK

This system is used to compare the B-schema with the T-schema and provides a database schema that supports both the existing and new applications. The input to this system is T-Schema and B-Schema and the output is the integrated database schema.

Notations:

T_i : an entity in T-Schema, $i = 1, 2, \dots, m$.

B_p : an entity in B-Schema, $p = 1, 2, \dots, n$.

$Attr(E)$: a set of attributes in entity E .

$R(E_1, E_2, \dots, E_n)$: a relationship participated by E_1, E_2, \dots, E_n .

$E_1 = E_2$: $Attr(E_1) = Attr(E_2)$, E_1 and E_2 are equivalent.

$E_1 \subset E_2$: $Attr(E_1) \subset Attr(E_2)$, E_1 is a subclass of E_2 .

$E_1 \supset E_2$: $Attr(E_1) \supset Attr(E_2)$, E_1 is a superclass of E_2 .

$E_1 \cap E_2 = \rightarrow$: $Attr(E_1) \cap Attr(E_2) = \rightarrow$, E_1 and E_2 are disjoint.

$E_1 \cap E_2 \neq \rightarrow$: $Attr(E_1) \cap Attr(E_2) \neq \rightarrow$, E_1 and E_2 are overlapping.

\rightarrow : denote “is-a” relationship

Algorithm:

begin

 Resolve the naming conflicts; /* similar to the step 1 in B-schema construction */

 Transform all relationships with degree > 2 into a set of binary relationships;

 Repeat

 For each relationship/entity in T-Schema
 compare and modify B-Schema by Rule 1

 Until there is no more evolution;

 For those entities haven't been added in B-Schema
 apply Rule 2 to integrate it;

end

The algorithm is complete because it checks all relationships and entities in both schema.

Figure 8a: A relationship in T-Schema*Figure 8b: A relationship in B-Schema*

Rules:

Rule 1: Comparison of relationships and entities

Without loss of generality, we can always transform a higher degree of relationship into binary relationships. Thus, we just consider the comparison of binary relationships here.

Let $RT = R(T_i, T_j)$, $i, j = 1, 2, \dots, m$, and $i \neq j$

$RB = R(B_p, B_q)$, $p, q = 1, 2, \dots, n$, and $p \neq q$

$T, T' \in \{T_i, T_j\}$, $T \neq T'$

$B, B' \in \{B_p, B_q\}$, $B \neq B'$

The representations in ER diagram are shown in Figure 8.

While comparing the relationships, we consider two cases:

i) $RT = RB$ (i.e. same relationship name).

RT can be usually eliminated by inheritance with entities have a sub-class/superclass relation or overlap from two schema.

ii) $RT \neq RB$ (i.e. different relationship name).

RT is usually added in B-Schema and cannot be eliminated in this case.

We will give a brief explanation in 1.1 as an example. Other cases are mostly similar and should be intuitive.


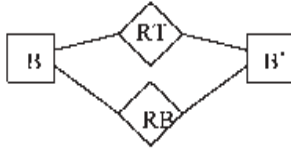
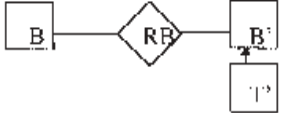
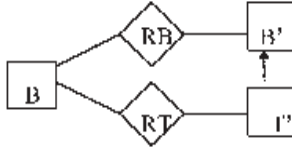
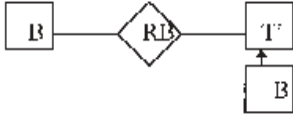
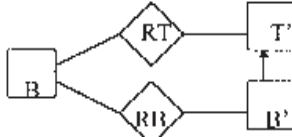
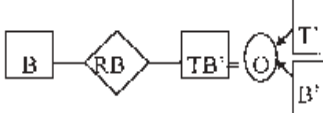
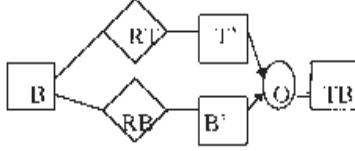
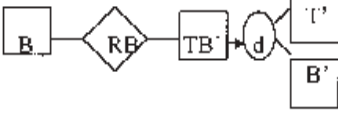
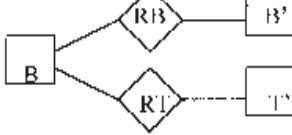
Rule 2: Handling the isolated entities

Users have to choose an entity in T-Schema for the combination in following ways.

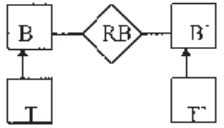
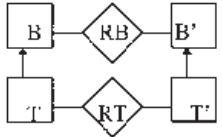
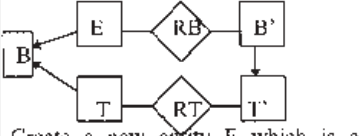
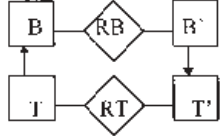
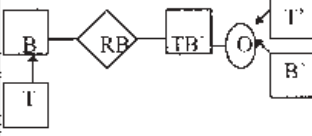
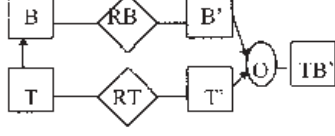
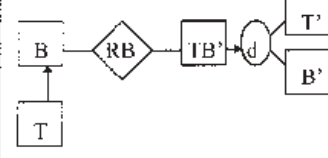
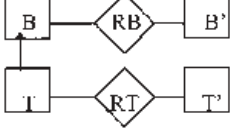
A CASE STUDY

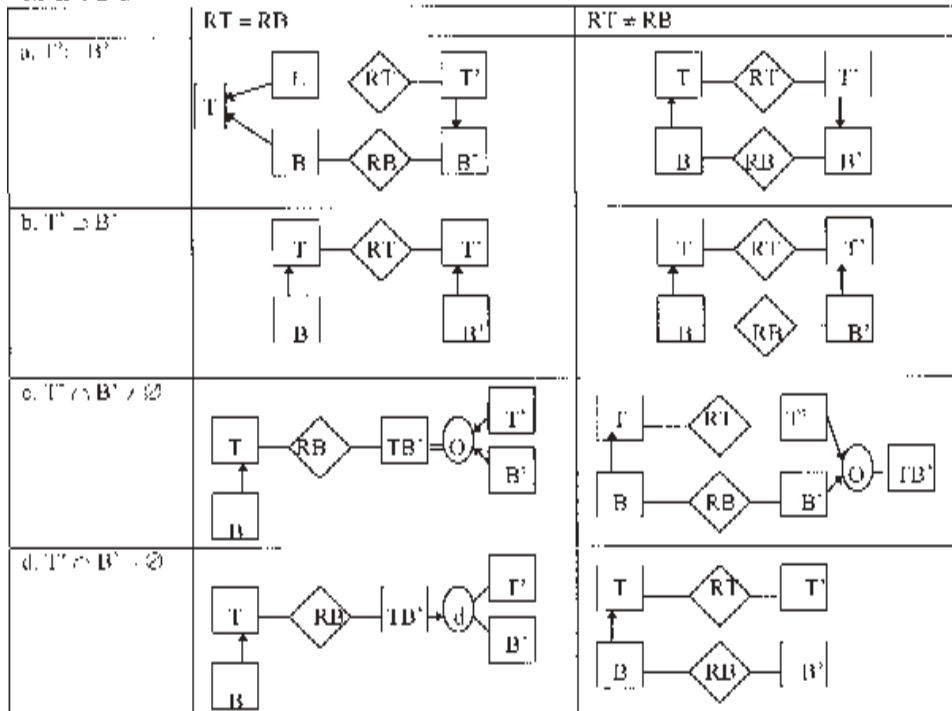
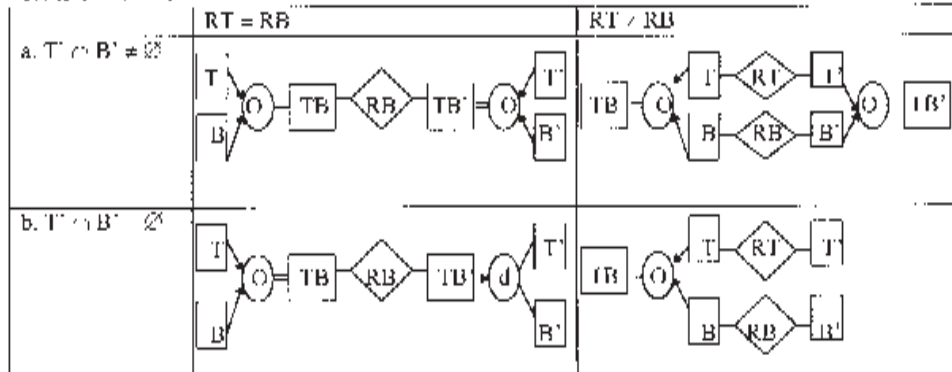
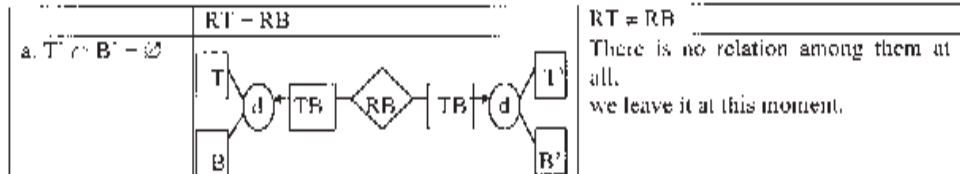
In a bank, there are existing databases with different schemas: one for the customers, another for the mortgage loan contracts and a third for the index interest

1.1 If $T = B$

	$RT = RB$	$RT \neq RB$
a. $T' = B'$	 <p>RT has been already contained in B-Schema.</p>	 <p>RT cannot be eliminated.</p>
b. $T' \subset B'$	 <p>RT is contained through inheritance.</p>	 <p>RT cannot be inherited.</p>
c. $T' \supset B'$	 <p>Similar to above.</p>	 <p>Similar to above.</p>
d. $T' \cap B' \neq \emptyset$	 <p>O = overlap generalization Since T' and B' overlap, we can make it as a generalization.</p>	 <p>Generalize T' and B'.</p>
e. $T' \cap B' = \emptyset$	 <p>d = disjoint generalization Since $RT \neq RB$, there must be some kind of relation between T' and B' though they are disjoint.</p>	 <p>For $RT \neq RB$, T' and B' are disjoint, we can treat it separately.</p>

1.2 If $T \subset B$

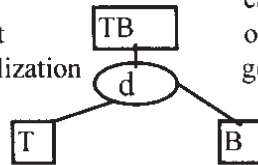
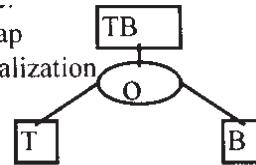
	$RT = RB$	$RT \neq RB$
a. $T' \subset B'$		
b. $T' \supset B'$	 <p>Create a new entity E, which is a subclass of B.</p>	
c. $T' \cap B' \neq \emptyset$		
d. $T' \cap B' = \emptyset$		

1.3 If $T \supset B$ 1.4 if $T \cap B \neq \emptyset$ 1.5 if $T \cap B = \emptyset$ 

2.1 Introduce a new relationship

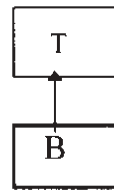


2.2 Introduce a new generalization

case 1:
disjoint
generalizationcase 2:
overlap
generalization

2.3 Introduce a subclass relationship

case 1



case 2

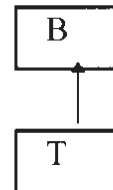
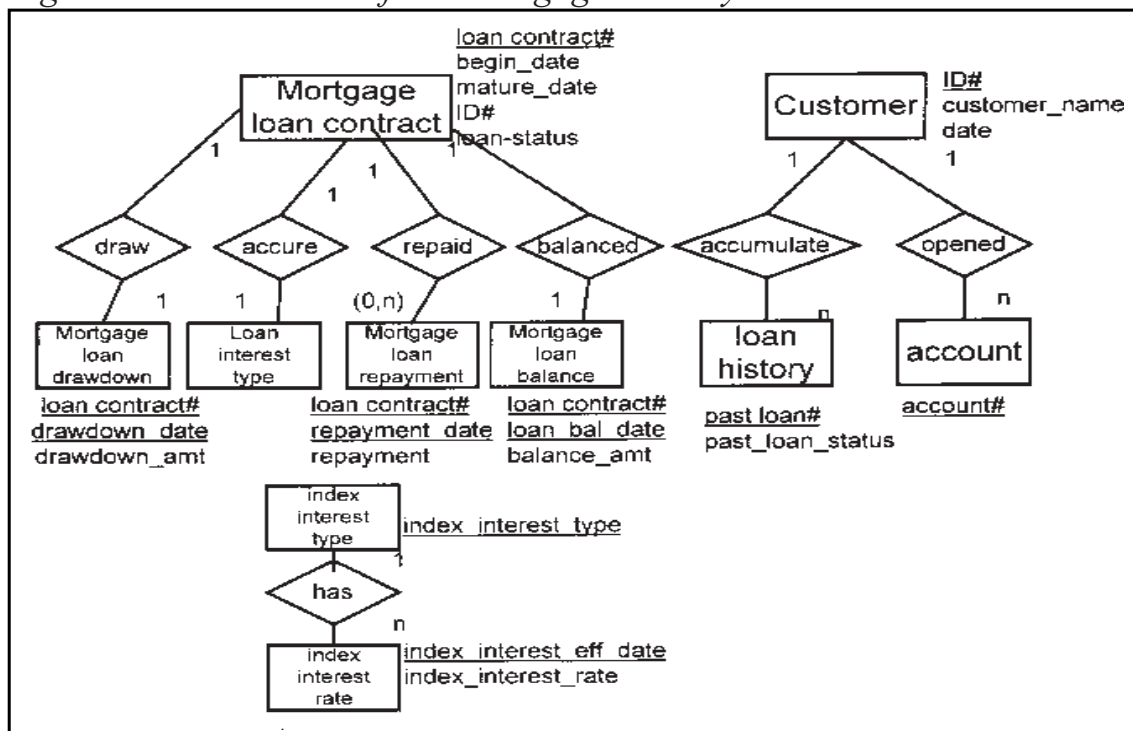


Figure 9: EER models of the Mortgage Loan System



rate. They are used by various application in the bank. However, there is a need to integrate them together for an international banking loan system. The followings are the three source schemas shown in Figure 9.

Step 1. B-Schema Analysis.

By following the methodology, in the first iteration, the mortgage loan schema will be merged with the customer schema in the first iteration as follows:

Figure 10: Integrated B-schema for mortgage loan system

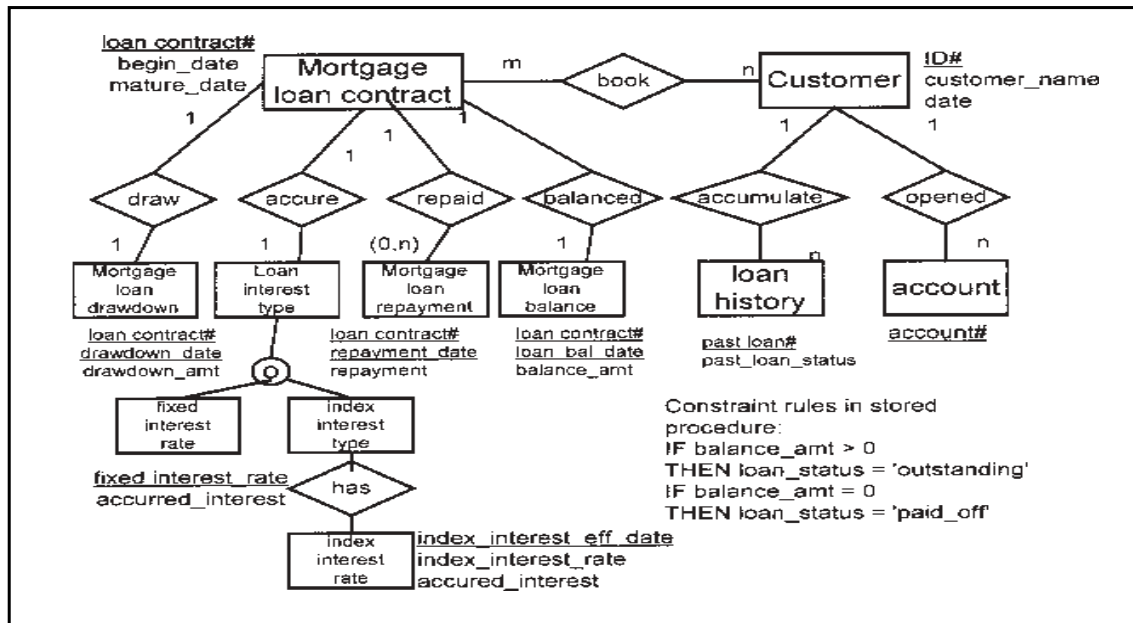
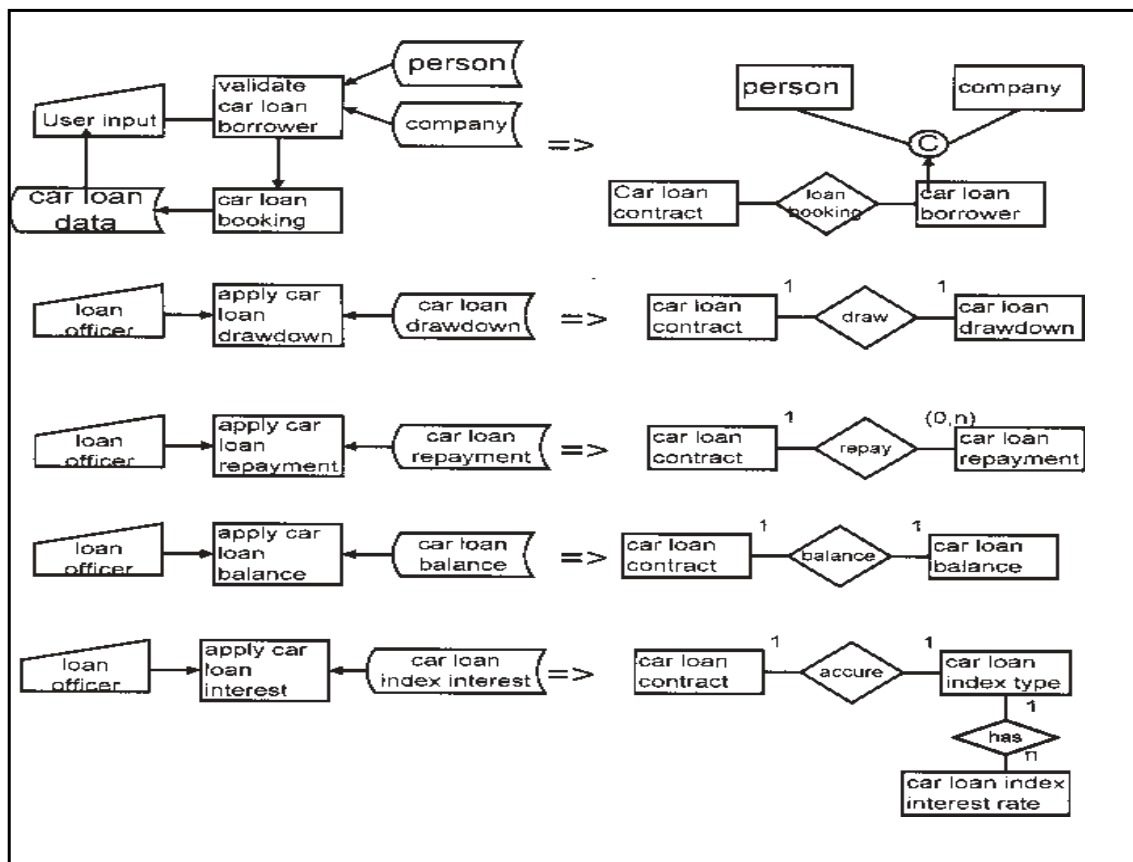


Figure 11: T-schema sub-processes and their data analysis



- There are two synonyms: Loan_status and Balance_amt such that the Loan_status can be derived from the Balance_amt. As a result, we can replace Loan_status by Balance_amt with a stored procedure to derive Loan_status value from Balance_amt.
- There is an implied relationship between these two schemas such that ID# used as attribute in loan
- schema but as an entity key in customer schema. Thus, we can derive cardinality from the implied relationship between these entities, and integrate the two schemas into one EER model.

In the second iteration, the intermediate integrated schema will be merged with the index rate schema. There is a overlap generalization between the two schemas such that a loan must be on fixed and index interest rate. Thus, by joining the integrated schema and the index rate schema with overlap generalization, the two schemas can be integrated in a B-schema in Figure 10.

Step 2. T-Schema analysis

2.1 Process analysis

T-Schema from the joint data and functional analysis involving the data flow diagram captures the new application process requirements and the data analysis takes care of the data requirements for the new application. In this case study, here is a functional bank car loan system. There is one main process of car loan processing which can be decomposed into five sub-processes in the loan system: car loan booking, car loan index rate update, car loan repayment, car loan balance and car loan index interest type update as shown in Figure 11.

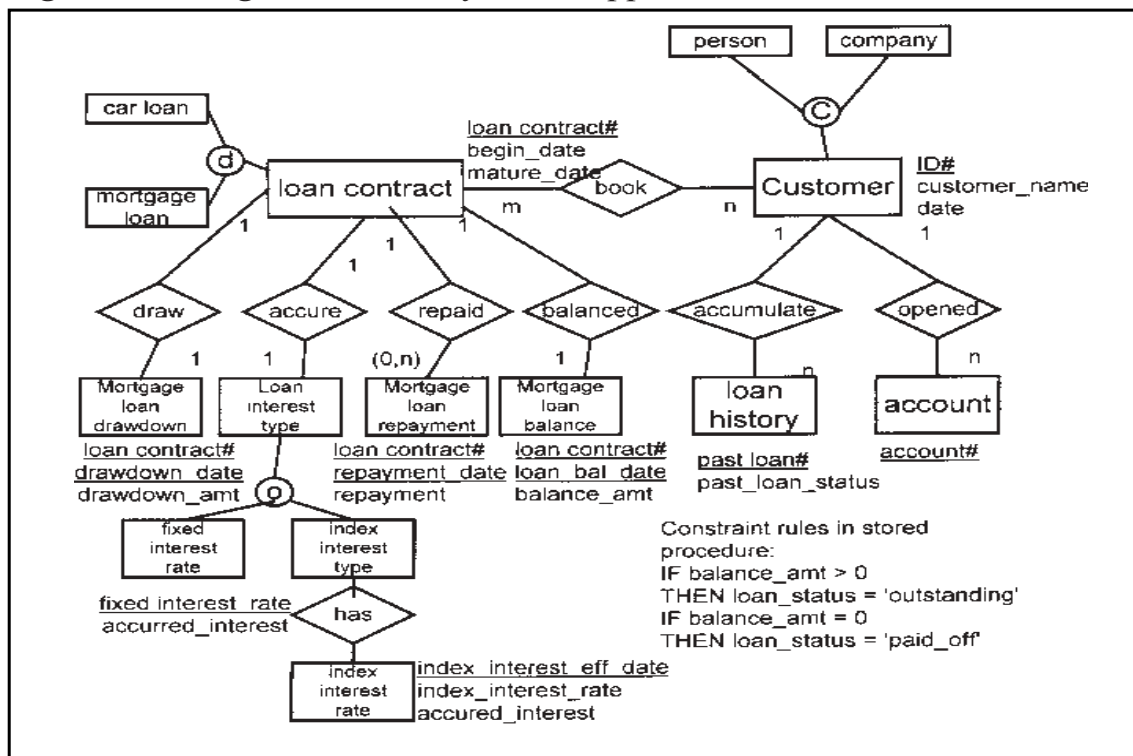
2.2 Data analysis

After the refinement of the functional schema, the refined data schema can be shown in Figure 12.

Step 3. Mutual completeness check.

The derived data schema contains a new data requirement of overseas customer, but does not contain a account record with respect to the B-schema. After the comparison between the data schema and the B-schema, we need to solve synonym conflict such that customer and car loan borrowers are synonyms, and the date of customer and the date of the car loan borrower are homonyms as customer_record_date and car_registration_date in Figure 13.

The refined B-schema contains the existing data requirements with Account record plus a new data requirements of different kinds of customer



and two disjoint loans: mortgage loan and car loan, as shown in Figure 14.

CONCLUSION

A mutual consistency checking methodology has been presented to integrate existing databases to support new applications. The approach is featured with a combined methodology which uses both traditional bottom-up method, and top-down method. The rationale behind such a combined methodology of conducting database schema integration is the fact that top-down approach can complement the bottom-up approach of schema integration by taking into account both data and function requirements of new applications. Thus the integrated schema can readily support new applications while continuing to serve existing ones.

It is important to extend the scope of conflict resolution performed by the mutual consistency checker. When the difference between the T-schema, and B-schema is more significant, the system may need to evolve the integrated schema and propagate the changes back to the individual local schemas. Likewise, the T-schema enables us to filter out unused ones from the integrated schema; a view mechanism is useful here and can be developed on top of the integrated schema. For further research, the feasibility of automating much of the schema analysis and integration process should be investigated by developing a set of heuristic rules which can be utilized by the expert system component.

REFERENCES

- Batini, C., Ceri, S. and Navathe, S.(1992) Conceptual Database Design: An Entity-Relationship Approach, *The Benjamin/Cummings Publishing Company, Inc*, 227-243.
- Batini, C., Lenzerini, M. and Navathe, S.(1986) A Comparative Analysis of Methodologies for Database System Integration, *ACM Computing Survey*, Vol 18, No 4.
- Elmasri R. and Navathe, S.(1989) *Fundamentals of Database Systems*, Benjamin/Cummings pub.
- Fong, J.(1992) Methodology for schema translation from hierarchical or network into relational, *Information and Software Technology*, 34(3),159-174.
- Fong, J., Karlapalem, K. and Li, Q.,(1994)A practitioners approach to schema integration for new database applications, *Proceeding of the 5th International Hong Kong Computer Society Database Workshop*, 136-154.
- Korth, H. and Silberschatz, A.(1991). *Database System Concepts* (2nd edition),

McGraw-Hill.

McLoed, D. and Heimbigner, D.(1980) A Federated Architecture for Database Systems, *Proceedings of the AFIPS National Computer Conference*, vol 39, AFIPS Press, Arlington, VA.

Ozkarahan, E.(1990)*Database Management: Concepts, Design and Practice*, Prentice-Hall.

Özsu, M. and Valduriez, P.(1991).*Principles of Distributed Database Systems*, Prentice-Hall International Edition, p428-430.

Senn, J.(1989).*Analysis & Design of Information Systems* (2nd edition), McGraw-Hill.

Sheth A. and Larson, J.(1990). Federated Database Systems for Managing Distributed Heterogeneous, and Autonomous Databases, *ACM Computing Survey*,22(3).

Ullman, J.(1982). *Principles of Database Systems* (2nd edition), Computer Science Press.

Chapter 12

CMU-WEB: A Conceptual Model for Designing Usable Web Applications

Akhilesh Bajaj and Ramayya Krishnan
Carnegie Mellon University

With the ubiquitous availability of browsers and internet access, the last few years have seen a tremendous growth in the number of applications being developed on the world wide web (WWW). Models for analyzing and designing these applications are only just beginning to emerge. In this work, we propose a 3-dimensional classification space for WWW applications, consisting of a degree of structure of pages dimension, a degree of support for interrelated events dimension and a location of processing dimension. Next, we propose usability design metrics for WWW applications along the structure of pages dimension. To measure these, we propose CMU-WEB: a conceptual model that can be used to design WWW applications, such that its schemas provide values for the design metrics. This work represents the first effort, to the best of our knowledge, to provide a conceptual model that measures quantifiable metrics that can be used for the design of more usable web applications, and that can also be used to compare the usability of existing web applications, without empirical testing.

Over the last five years, there has been a tremendous growth of applications being developed to run over the world wide web (WWW) (Berners-Lee, Caillau, Luotonen, Nielsen, & Secret, 1994). Several technologies are in vogue for writing these applications (Bhargava & Krishnan, Forthcoming).

Depending on the kind of technology used, different classes of applications can be created using the WWW as the medium of transport.

Given the large number of systems analysis and design methods available, there is some confusion as to which methods are suitable for WWW applications. This work makes two contributions. First, we present a three dimensional classification space for WWW applications. The dimensions used are the *location of processing*, the *degree of support for interrelated events*, and the *structure of pages*. The classification scheme we use provides insight into which existing modeling methodologies are useful for designing a WWW application along each dimension. We find that adequate models exist for the *location of processing* and the *interrelated events* dimension. However, while several modeling methods (e.g., (Bichler & Nusser, 1996; Isakowitz, Stohr, & Balasubramanian, 1995)) have been recently proposed for the documentation and maintenance of WWW applications, there is a need for a conceptual model that can facilitate the design of WWW applications along the *degree of structure of pages* dimension, such that the applications are more usable. We propose design criteria that relate to usability, and hold along the *structure of pages* dimension, in the form of high level requirements. These requirements represent questions that should be answered by a conceptual model that seeks to facilitate the design of WWW applications so that they are more usable.

The second contribution of this work is a model that solves the above need to be able to quantitatively evaluate the high level requirements. We propose CMU-WEB (Conceptual Model for Usable Web Applications), a conceptual model of WWW applications that facilitates the design of more useable WWW applications, by providing a schema which provides values for metrics that measure the high level requirements along the structure of pages dimension.

The rest of this paper is organized as follows. First, we present a 3-dimensional classification space for WWW applications. Then, we propose a list of high level usability metrics, along one dimension of the space. Next, we define CMU-WEB, and show how usability metric values can be derived from CMU-WEB schema. Finally, we give directions for future research and the conclusion.

A CLASSIFICATION SPACE FOR WWW APPLICATIONS

In this work, we define a WWW application as one that runs using the hypertext transfer protocol (HTTP) as the transfer protocol. In our view, this

is what differentiates WWW applications from other networked applications. We define an application as consisting of a series of zero or more events. We define an event as a subset of an application that consists of at least one user input, followed by some processing.

Networked applications in general differ along several dimensions, such as the degree of state maintained on the server, the class of user, the type of user interface and the programming language used. Before identifying dimensions for classifying WWW applications (which are a subset of all networked applications) we identify certain features that are shared by all WWW applications:

- All WWW applications are inherently client/server. The WWW client is a web browser, which communicates with a WWW server using HTTP¹.
- The HTTP protocol is inherently stateless², which means that the server does not maintain the state of the client's application. However, several recent WWW applications involve the downloading of a client (e.g., a chat client) that then establishes a stateful link with its corresponding server (e.g., a chat server).
- The bulk of processing is usually done on the server side, although this is not necessary any more.
- The direction of data is two-way. Multimedia data³ flows from the WWW server(s) to the client, while alphanumeric data⁴ flows from the WWW browser to the WWW server.
- A large percentage of WWW applications are accessed by heterogeneous, naïve users.

These features are common to all WWW applications. They are also what makes WWW applications different from networked applications running on some other protocol, such as CORBA⁵ (Common Object Request Brokered Architecture) or RPC (Remote Procedure Call).

Next we propose three dimensions along which WWW applications differ: the degree of structure of the WWW pages in the application, the location of processing and finally, the degree of support for interrelated events within an application. WWW applications can be classified along several different dimensions, such as the technology used in building the application, whether the application is transactional or not, or whether the groups accessing the application are naïve or expert. The three dimensions that we propose here serve the purpose of providing insight into the role of different design methodologies that are appropriate for constructing a WWW application.

Degree of Structure of the WWW Pages

This dimension looks at the degree of structure of the pages that make up a WWW application. Values along this dimension include pages following

the Hyper Text Markup Language (HTML), pages following the Extensible Markup Language (XML) (Khare & Rifkin, 1997) and pages with completely flexible content, determined by the application. We now explain each of these values.

Most WWW pages today are in HTML format. An HTML page presents a hypertext interface to the user. HTML tags do not convey any meaning as to the structure of the contents of the page. WWW clients simply interpret the tags as display instructions.

The second value along this dimension is XML format. XML is an emerging standard for client browsers, and is a subset of the Standard Generalized Markup Language (SGML). XML allows the definition of the structure of the page using tags that the creator of the document can define at will. It is hoped that using tags for purposes other than merely display⁶, as in HTML, will solve many of the problems that plague HTML pages. E.g., A group of organizations could agree on a set of tags that convey the same content. This will facilitate the development of applications that share information across these organizations, by greatly reducing the amount of procedural code that would need to be written to create structure from a flat HTML format.

The third value along this dimension is complete flexibility of user interface. This is now possible by using Java applets that allow a browser to download a document that contains information that allows the execution of a Java applet (Arnold & Gosling, 1996). The applet is stored on the WWW server in bytecode format, and executed on the client machine. The applet can be used to create interfaces that are completely flexible. E.g., Different chatroom applets on the WWW present different interfaces to users. Complete flexibility allows pages to be structured, and presented in any way desired.

Next, we discuss the second dimension: the location of processing.

The Location Of Processing Of The WWW Application

This dimension looks at whether the processing (if any) takes place on the server side, or on the client and server side. Hence we have four values for this dimension: no processing, processing only on the server, processing only on the client and processing on the client and server. We now explain each of these values.

A large percentage of WWW pages are used for information display only. A WWW application with no processing would consist of a list of linked WWW pages. Note that processing would still be necessary for following the HTTP protocol on both client and server side. However, there is no processing of content.

Processing only on the server arises because of the Common Gateway Interface (CGI) (net.genesis & Hall, 1995). The interface, allows a browser to download a WWW page, and then to submit alphanumeric data to a WWW server. The WWW server receives the data and passes it to a CGI script. The data is passed using environment variables on UNIX systems, and temporary files on Windows-NT systems. The processing program, which can be in any programming language, reads this data and processes it. The results are passed back to the client, either directly by the program or via the WWW server. The result is usually another page, perhaps generated dynamically. Note that no processing takes place on the client side here. WWW applications using HTML forms that require user input use CGI.

Processing only on the client involves client-side scripting or Java applets or Active X controls. In client side scripting, the page includes programs in an interpreted language such as *Javascript* or *Vbscript*. e.g., the function is coded in the HEAD of an HTML page, and then accessed in the BODY of the page. There are a few problems with using client side scripting for large amounts of processing (Bhargava & Krishnan, Forthcoming). First, the script is interpreted, and is slower than compiled programs that usually run on the server side. Second, the source code is available to the client, which may be undesirable. Third, increase in size of client side scripts causes slower downloads. In general, light processing like error checking input is performed using client side scripting.

Active X controls or Java applets also allow client side processing. The downloading and execution of an applet in the browser allows a larger amount of processing to be handled by the client than is the case with client-side scripting. Note that if Java applets are used, it is possible to bypass HTTP, and to establish a persistent state network connection using Java's extensive networking support. Typically, Java applets and Active X controls are used to create user-interfaces. However, they can also be used to perform more processing on the client side.

Processing on both the client and server means that the processing of events in the application is divided between the client and the server. This involves the use of CGI (for processing on the server) and of client-side scripting or Java applets or Active X controls (for processing on the client).

Next, we discuss the third dimension: the degree of support for interrelated events within an application.

The Degree Of Support For Interrelated Events

This dimension measures the degree to which the events within the WWW application can be interrelated to each other. We propose 4 values

along this dimension: no events, only independent and idempotent (I&I) events, sets of I&I events interspersed with interrelated events and sequences of interrelated events. We now explain each value.

No events occur in applications with an absence of processing of any content. This would happen in an application that simply displayed pages, and allowed for hypertext navigation between pages. This is also called a kiosk application (Troyer & Leune, 1998).

Events processed on WWW servers are I&I events because of the stateless nature of HTTP, i.e., the server can not keep track of events in the application⁷. E.g., if a CGI application requires the client to supply a series of forms that are written to server files, then each time the “submit” button is pressed on the client, an application event is generated on the server. Since HTTP is stateless, each “submit” event from the client is treated without knowledge of any previous submits. There is no way to keep track of the state of how many write-events have been done by a client, or whether a client decided to repeat some write-events by resending some forms of the application⁸.

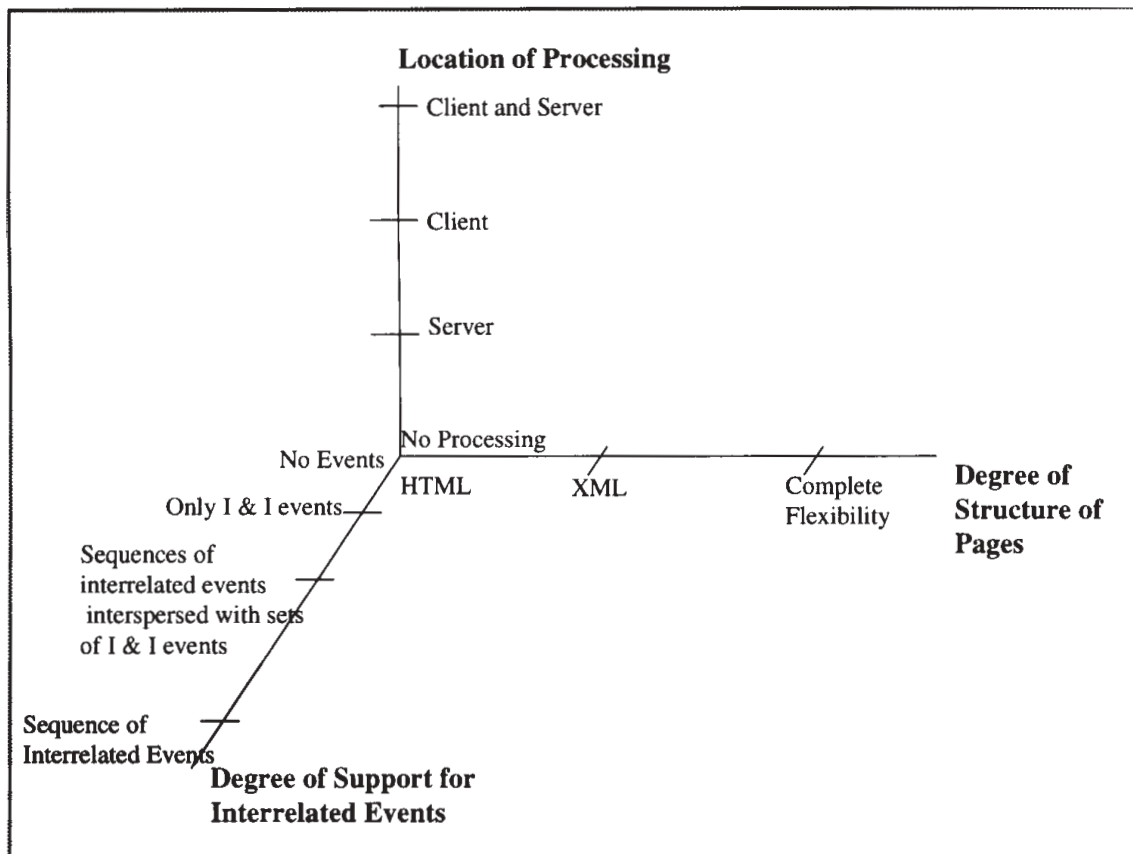
In a well-designed WWW application of this type, the events that are generated on the WWW server should be *idempotent* (each event in an application instance can be run multiple times without changing the outcome of the application instance). Also, server events should belong to event sets, i.e., there should be no interdependence between the different events in the application, represented by different pages. This is needed because it is impossible to keep track of the state of the application instance between events. therefore, in an application of this type, the only solution is to clump all inter-dependent input from users in an application on one page, as one event.

Sets of I&I events interspersed with sequences of interrelated events arise in the case of processing on the client and server side, where the client is a browser, and the server is the WWW server. Note that the client can maintain state of the application. Thus, in a WWW application of this type, interrelated events are processed on the client side, and I&I events are processed on the server side⁹. This kind of application will consist of a sequence of interrelated events (on the client), followed by a set of server (I & I) events, followed by another sequence of client events, etc. An example of this would be performing error checking on an HTML form for correct format masks, permissible value ranges, etc by a client side script and then sending the form to the server. The checking at the client side leads to a sequence of interrelated events, written as a client side script. The submission of the form to the server leads to an I & I event.

Sequences of interrelated events arise in the case of processing on the client and server side, where a special client (e.g., a chat client) can be downloaded on a WWW browser, and can establish a stateful link with its corresponding (chat) server. Once a stateful link is established, the application becomes a sequence of fully interrelated events, since both the (chat) client and the (chat) server can keep track of the state of the (chat) application. WWW applications that employ state maintenance technologies like cookies can also contain sequences of interrelated events.

Our three dimensional space for classifying WWW applications is shown in figure 1. An application is classified by a triple that represents values along the three axes. E.g., a WWW application using a HTML pages, with *Javascript* and CGI sharing the processing would be (*HTML, Client and Server, Sequences of interrelated events interspersed with I & I events*). A WWW application that displayed HTML pages would be (*HTML, no processing, no events*). A WWW chat room application that involved downloading a Java applet chat client that connected to a chat server would be (*complete flexibility, client and server, sequence of interrelated events*).

Figure 1. 3-d Space for Classifying WWW Applications



Insights from the Classification Scheme

The classification scheme provides insight into what models should be used in the analysis and design phase of a WWW application. First, for the *degree of support for interrelated events* dimension, no model that depicts events is needed for the no events value. In case of the other three values on the dimension, well known systems analysis and design techniques such as Object Oriented Analysis and Design (OOAD) (Booch, 1994) can be used. Techniques like OOAD are very suitable for applications which are a sequence of interrelated events. They do not however, to the best of our knowledge, allow the modeling of the idempotency of each event or a series of events. This is an area for future research. Apart from this, OOAD or other methodologies can be used to design applications along all values of the degree of support for interrelated events dimension, except for applications with no events. There is a large body of literature on using well known systems analysis and design models with some metric based feedback to assess the quality of a good design (e.g., (Booch, 1994; Bulman, 1991; Jefferey & Stathis, 1996; Martin & Odell, 1992)). Examples of well known metrics for this dimension include lines of code, the function point metric and high level requirements include the sufficiency, primitiveness and completeness of methods and the coupling and cohesion of classes in OOAD.

Second, for the *location of processing* dimension, no models that allow the analysis of sharing of processing are needed for no processing¹⁰. Also, no models are needed for WWW applications where all the processing is done on one machine (only on the WWW client or only on the server). There is a substantial body of literature on designing client/server applications, where processing is distributed between the client and the server (e.g., (Boar, 1992; Deshpande, Jenkins, & Taylor, 1996; Drakopoulos, 1995; Major, 1996)). The design methodologies in this literature also provide some metric based feedback on what a good design is. Many of these design methodologies involve creating discrete event simulation models of the client server interaction (Deshpande et al., 1996) or analytical queuing models (Drakopoulos, 1995). Examples of well known metrics along this dimension include CPU utilization of client and server, disk input/output (I/O) utilization of client and server, average wait time for client requests at server and average run queue length at the server.

Third, for the *structure of pages* dimension, several models have recently been proposed on how to document HTML applications (e.g., (Bichler & Nusser, 1996; Isakowitz et al., 1995; Schwabe, Rossi, & Barbosa, 1996)). To the best of our knowledge, most models that have been proposed to model

WWW applications actually model only (*HTML, no processing, no events*) WWW applications. As we point out in (Bajaj & Krishnan, 1998) these existing models do not provide any metric based feedback on how good the *design* of the application is; instead, they focus on *documenting* the application and facilitating easier *maintenance*. Several metrics have been identified for hypertext applications (Botafofo, Rivlin, & Shneiderman, 1992) as well as for user-interfaces in general (Nielsen, 1993; Preece, 1994; Shneiderman, 1998). Next, we draw from this previous work and identify a set of high level metrics that should be supported by a conceptual model that provides metric based feedback on how good a WWW application is along the *structure of pages* dimension.

USABILITY REQUIREMENTS ALONG THE STRUCTURE OF PAGES DIMENSION

Several high level requirements have been proposed for hypermedia documents (see (Garzotto, Mainetti, & Paolini, 1995) for an extensive listing). We present what we view as a canonical set of four quality-requirements, i.e., the set covers most of the abstractions covered in quality requirements proposed by others, and each requirement in the set is reasonably non-overlapping with the other. The high level requirements we present in this section represent design questions that should be answered by a conceptual model that aims to facilitate the usability design of a WWW application. In an upcoming section, we present CMU-WEB, a conceptual model that answers these questions.

The first two quality requirements attempt to measure the readability of the HTML and XML documents in the application. Two factors influence readability: *coherence* as a positive influence (Thuring, Hannemann, & Hake, 1995) and *cognitive overhead* (Conklin, 1987) as a negative influence.

Coherence

This requirement is used to represent the ease with which readers can form mental models of a possible world, from the hypertext document. *Local coherence* is the degree to which each specific document conveys a mental model. E.g., A document that uses conjunctions, paragraphs and other writing techniques, with appropriate multimedia illustrations provides higher local coherence than one that simply contains free flowing text. *Global coherence* deals with the “lost in hyperspace” problem and is the degree to which the reader can form a macro structure across documents, e.g., an application that

maintains a preservation of context across nodes (perhaps by using HTML frames) is likely to be more globally coherent than one whose documents are disjoint fragments of context, with no cues in each document as to the pre-context or the post-context. An important determinant of global coherence is the *difficulty of navigation*, e.g., an application that does not permit backward navigation, or that has arbitrary jumps from each page is less easy to navigate than one that supports navigation in both directions, and that has a smaller number of jumps, with less “cognitive distance” per jump.

Cognitive Overhead

The reason for this requirement comes from the limited capacity of human information processing. In hypermedia applications, cognitive overhead is determined primarily by *user-interface adjustment* (Thuring et al., 1995) and *low consistency* (Garzotto et al., 1995). For example, an application that presents too many or changing fonts, colors and layouts on each page requires more user interface adjustment than one that presents a uniform appearance between pages with fewer fonts and colors and the same layout. An application that depicts say, sound multimedia objects differently in different pages is less consistent, and would impose greater cognitive overhead on the reader.

The previous requirements related to HTML and XML values along the dimension, since they are based on the fact that the interface for these structures is hypertext.

The next two requirements relate to the actual information presented on the pages of a WWW application.

Cohesion Of Information In A Document

This requirement represents the need that information in a single page is cohesive in the real-world it represents. E.g., if the page contains information on customers alone, then it is more cohesive, and hence better, than a page that contains information on customers as well as sales.

Coupling Of Information Across Documents

This requirement represents the need that information in a page should be independent of information in other pages in the application. E.g., if the customer name and address are duplicated across pages in the application, the coupling will be more, and hence the application will be of lower quality, than if only a key field like the customer number is duplicated across documents¹¹.

The following high level requirement pertains only to the *complete flexibility* value on the dimension. It measures the usability of a non-hypertext interface.

Usability Of Non-hypertext Interfaces

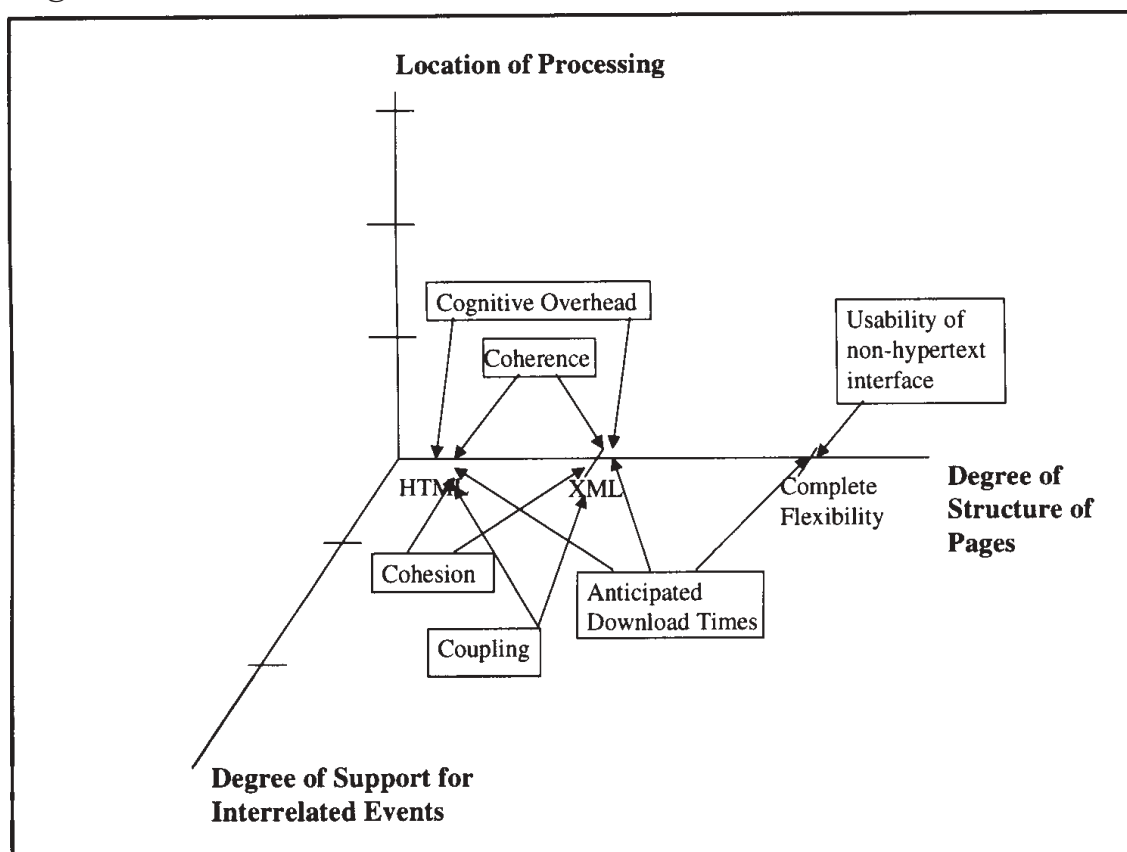
There is a great deal of work on the usability of human interfaces (e.g., (Nielsen, 1993; Preece, 1994)). We define the following factors as influencing the usability of user interfaces. These factors are borrowed from (Nielsen, 1993), and represent a synthesis of over 90 published studies in the area of user interface design. The *learnability* of the interface is the perceived ease of use of the application by novice users. *Efficiency of use* is the steady-state performance, once the learning curve flattens out. *Memorability* of the interface is the amount of time that elapses before users slip back on the learning curve.

The sixth high level requirement comes from the fact that all applications run on a network, and that network delays, slow servers, etc. can lead to long download times. It is hence applicable to all values along the dimension.

Anticipated Download Times

This requirement represents the time it will take to download pages from the WWW server. It is determined by endogenous factors like the *server*

Figure 2. Usability Requirements for values along the Degree of Structure of Pages Dimension



capacity, the *size of the objects* that would appear on each page of the application, and the *number of different HTTP requests* that need to be sent to create each page of the application. It is also determined by exogenous factors like *the network traffic* at the time of download and the *number of requests to the WWW server*. Applications on servers with greater processing and disk input/output capacities, with smaller size objects on each page and requiring fewer HTTP requests per page are likely to have less download times.

The high level requirements are summarized in figure 2. Note that the requirements we have proposed here stem from our classification scheme, and are for WWW applications, not for models of WWW applications. These requirements represent information that should be derivable from schemas of models that are used to design WWW applications along the structure of pages dimension.

Next, we propose CMU-WEB: a conceptual model for usable web applications that seeks to facilitate the design of WWW applications, by providing metrics that evaluate the fulfillment of requirements identified in this section.

CMU-WEB: CONCEPTUAL MODEL FOR USABLE WEB APPLICATIONS

Components and Semantic Rules of CMU-WEB

We define a CMU-WEB schema as a graph with nodes and arcs. The **node** types are:

1. **Canvas View:** A canvas view (CV) is a full page in a WWW application, with at least one information chunk (defined below). Examples of canvas views are: HTML pages, XML pages, Java forms and subsets of HTML pages if they are anchored (pointed to by a link). A WWW application is structured as a network of canvas views, with hyper links between them.
2. **Information Chunk:** An information chunk (IC) is a clearly discernible discrete unit of information that is represented in the application. Examples of ICs include: *a textual paragraph describing an idea, an attribute* (say, customer name) of a real world entity (say, customer), a *.gif* file; a *.jpeg* file and *textual information* about a real world entity. Only text based information can be split into smaller ICs. A photograph, a movie or a piece of sound are considered to be single ICs. The problem of splitting these multimedia files into smaller ICs is for future research.
3. **Information Chunk Depiction:** An information chunk depiction (ICD) is the method of depicting the IC in the application. The same IC may have

several different depictions. E.g., A map to a location (*.gif*) and verbal directions to the location (*text*).

4. **Hyperlink Within Application:** A hyperlink to within application (HLWA) is a hyperlink from one CV to another CV in the application. HLWAs can be one-way (HLWA1) or two-way (HLWA2: the CV that is pointed to, also points back to the original CV). HLWA2s are implemented as two separate links in a WWW application.
5. **Hyperlink Outside Application:** A hyperlink outside application (HLOA) is a hyper link to a CV of another application.

The **arc** types are:

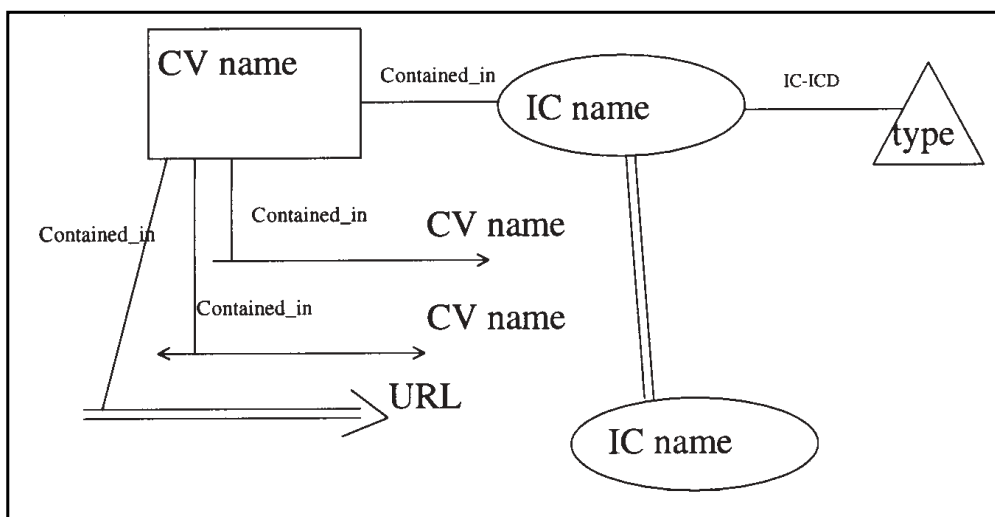
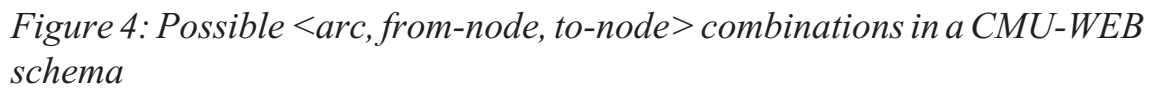
1. **Relationship Between Information Chunks:** A relationship between information chunks (RIC) is any **semantic** relationship between two ICs, that is deemed important for the potential users of the application. Examples of RICs include *both_work_for_MIS* as an RIC between *information_on_bob* and *info_on_mary*, and *both_introduce_company* as an RIC between a *text paragraph describing the company* and a *photograph of the company location*. We note that RICs differ from hyperlinks, in that they are conceptual and between ICs, while hyperlinks are actual and used for navigation between CVs.
2. **IC-ICD :** This arc type simply connects the ICD with the relevant IC.
3. **Contained-in:** This arc type connects an IC, an HLWA1, an HLWA2 or an HLOA with each CV that it appears in.

Informal rules for a CMU-WEB schema

A WWW application is one that consists of at least one Canvas View, and at least one Information Chunk¹². A CMU-WEB schema shows the RICs between all information chunks, as well as the canvas view locations of all information chunks, HLWA1, HLWA2, HLOA.

The graphic representation of components in CMU-WEB is shown in figure 3. The possible <arc, from-node, to-node> combinations in a schema are shown in figure 4.

Several other models use one or more of the concepts in CMU-WEB. For example, a widely accepted model of hypertext is a graph of nodes and links. The nodes there are the same as CVs here, and the links are the same as HLWA1s. Most models for developing WWW applications use the concept of a page, which corresponds to a CV here. The IC, ICD and RIC concepts in CMU-WEB are different from other models, but most *models* use some sort of primitive for capturing the information on a page. E.g., RMM (Isakowitz et al., 1995) uses a slice as that portion of a real world entity that appears on a page. The main difference between an IC and a slice is that an IC is not



restricted to belonging to some entity, and there is no need in CMU-WEB to aggregate all the chunks of information into entities, before slicing them into pages. This allows the depiction of information like abstract ideas, which may not be easily aggregated into entities. It also eliminates the creation of multiple models, and reduces the time taken to arrive at a schema. CMU-WEB is similar to the simple entity relationship model (Chen, 1976) which consisted of very few components, and where the identification of entities and relationships allows for great flexibility. When creating a CMU-WEB schema, the identification of what a relevant IC is, and what the relevant RICs are, is *highly application specific* and dependent on the designer of the application.

Another point of structural difference between CMU-WEB and most other current models for WWW applications is the *number of components*. Most other models contain significantly more components than CMU-WEB, and have significantly more semantic rules for consistency. This makes them more likely to be hard to use than CMU-WEB for modeling real-world applications (Castellini, 1998). For example, RMM has eleven components and several rules for consistency. CMU-WEB has six components (IC, RIC, CV, HLWA1, HLWA2, HLOA) that require cognitive effort to model.

Figures 5, 6 and 7 show a portion of a CMU-WEB schema that we created for an existing web application. These figures are meant to indicate what a CMU-WEB schema looks like. For convenience, the contained-in arc is assumed. Thus, all the elements in figure 5 are contained in the CV shown in figure 5, etc. Since there are no RICs between ICs contained in different CVs, we can do this here for convenience. In general, the contained-in arc should be shown, if RICs exist across CVs.

Next, we show how CMU-WEB can be used to derive values for the high level metrics, identified earlier, that indicate the usability of WWW applications.

Deriving Values for Usability Metrics From A CMU-WEB schema

Coherence

Local coherence deals with each CV. The local coherence per CV is increased if there are more RICs between the ICs on the CV. We measure this by the *local coherence due to RIC* (LCRIC) metric:

$$\text{LCRIC} = \frac{\sum \text{RIC}}{\sum \text{IC}} \quad \text{where the summation is over all the RICs and ICs on the CV}$$

Figure 5: One CV from a CMU-WEB schema created for an existing WWW application

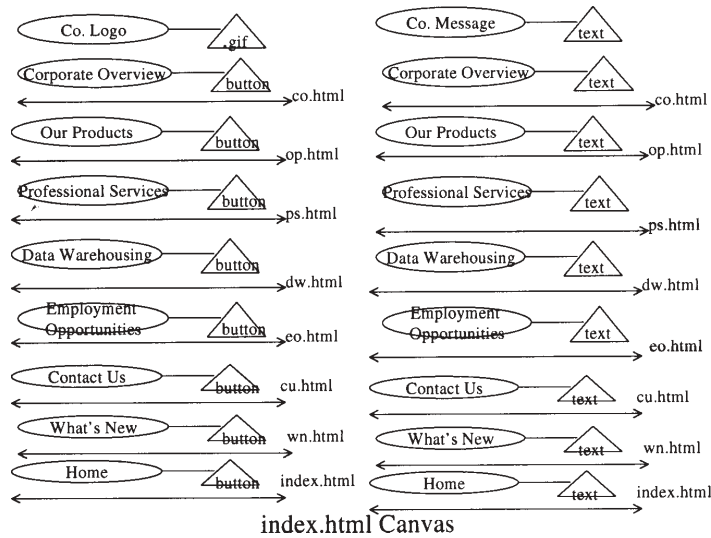


Figure 6: One CV from a CMU-WEB schema created for an existing WWW application

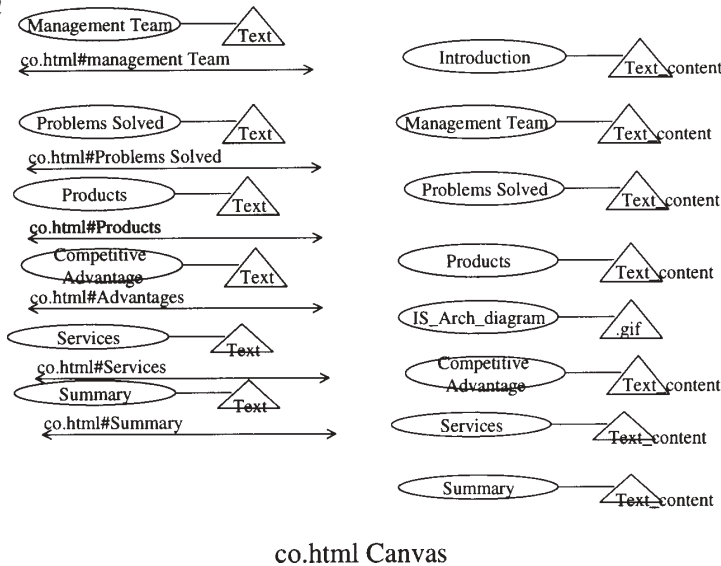
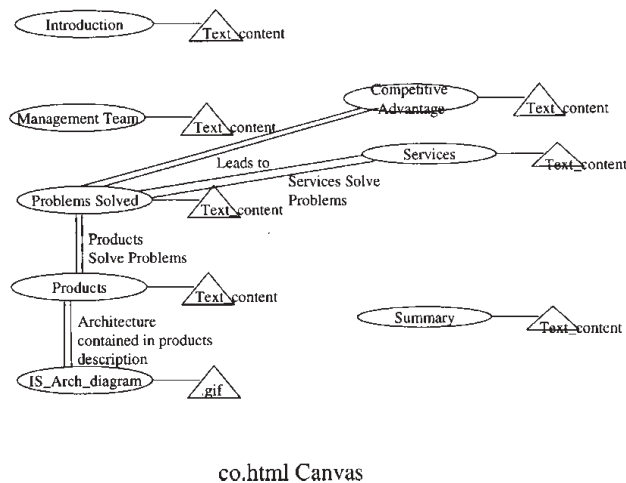


Figure 7: One CV from a CMU-WEB schema created for an existing WWW application



We can use the mean LCRIC, across all CVs in the application, as well as the standard deviation to measure the LCRIC in the application, as a whole. The range of LCRIC is 0 to infinity. A higher LCRIC indicates more local coherence. The mean LCRIC and its standard deviation (std. dev.) can be used to compare applications differing in number of CVs and HLWAs.

In addition to LCRIC, local coherence is also dependent on the well known principle that the capacity of the human short term memory is 7 ± 2 information chunks. Thus, a CV with more than 9 ICs will have less local coherence, while one with less than 5 ICs is being inefficient. We define a second metric for local coherence called the *local coherence due to short term memory* (LCSTM).

$$\text{LCSTM} = \frac{\sum \text{IC}}{\text{CV}} \quad \text{where the summation is the number of ICs across the page.}$$

This metric ranges from 1 to infinity, but the optimum values are between 5 and 9. We can also get the mean LCSTM across and the standard deviation, to measure the LCSTM in the application as a whole. The mean LCSTM and its standard deviation are comparable across applications of differing sizes.

Global coherence deals with the “lost in hyperspace” problem. Global coherence is higher if the number of HLWAs per CV is higher. We define a metric called *global coherence due to HLWA* (GCHLWA).

$$\text{GCHLWA} = \frac{\sum \text{HLWA1} + 2 * \sum \text{HLWA2}}{\sum \text{CV}} \quad \text{where the summation is across the application.}$$

GCHLWA ranges from zero to infinity. The higher the value, the more the global coherence. The metric is comparable across applications of differing sizes.

A second component of global coherence is the *difficulty of navigation*. (Botafogo et al., 1992) propose several low level metrics that deal with the structure of hypertext documents and permit easier navigation. Since they model a hypertext as a network of nodes and links, we simply note here that their metrics can be easily derived from a CMU-WEB schema, since a node = CV and a link = HLWA1.

We provide a guideline here based on the fact that a hierarchical structure is widely known to be easier to navigate than a network, as long as the number of levels is less than 4 (Vora, 1997). Given that the short term memory capacity of humans is between 5 and 9 information chunks, this translates to

a guideline that applications having less than $9*9*9 = 729$ CVs (other than pure menu based CVs) should use a hierarchical tree structure. Applications with more than 729 CVs should provide a search engine (Botafogo et al., 1992) or some sort of logical ordering of CVs.

A third component of global coherence is the *cognitive jump* between two CVs that are hyperlinked. Thus in each direction, an HLWA has a cognitive jump associated with it. This is dependent on the number of RICs between the source and the target CV, and also on the number of ICs in the source CV that are involved in these RICs. If there are more RICs between the source and target CVs, the cognitive jump is lower. However, the jump is higher the more the number of ICs in the source CV that are involved in these RICs. Thus, an ideal case would be where there are a large number of RICs between the two CVs, but only one IC is involved in these RICs, on the source CV. We also note that the number of ICs involved at the source can at most equal the number of RICs between the 2 CVs.

To measure this, we define a metric called *global coherence due to cognitive jumps* (GCCJ), for each HLWA, along each direction.

$$\begin{aligned} \text{GCCJ} &= 2 && \text{if the number of RICs between the source} \\ & && \text{and target CV is 0} \\ &= \frac{\sum \text{IC}}{\sum \text{RIC}} && \text{where the summation is for each link,} \end{aligned}$$

In the GCCJ metric, IC represents the ICs in the source CV that are involved in the RICs, and RIC is the RICs between the 2 CVs. The range of GCCJ is between 0 and 2. The lower the value, the lower the cognitive jump. We can also compute the mean GCCJ across each direction of all HLWAs in the application, as well as the standard deviation. The mean GCCJ and its standard deviation are comparable across applications of differing sizes.

Cognitive Overhead:

Cognitive overhead consists of *user interface adjustment* and *consistency* of the application. User interface adjustment is best determined through empirical testing, since it is highly dependent on the kinds of colors, fonts, layouts used.

To measure the consistency of the application, we propose a metric called the *cognitive overhead due to consistency* (COC).

$$\text{COC} = \frac{\sum \text{ICD}}{\sum \text{Media different types of media}}$$

where the summation is across the

Thus, if the application uses 3 types of media (e.g., text, sound, video) and uses two different formats for sound, and one each for video and text, then the value of COC is $4/3 = 1.33$. The range of the COC metric is 1 to infinity, and the higher the value, the more is the cognitive overhead due to consistency. The COC metric is comparable across applications of differing sizes.

Cohesion

Cohesion looks at how interconnected the information on each CV is. To measure this, we define an IC cluster as the ICs on a CV that are reachable from each other, using the RICs as links. Each CV has a minimum of one IC cluster, and a maximum of infinite IC clusters. We define a metric called *cohesion* (COH) to measure cohesion

$$\text{COH} = 1 / \Sigma \text{ IC clusters} \quad \text{where the summation is across a CV}$$

The range for COH for each CV is 0 to 1. The higher the value, the more cohesive is the CV. We can compute the mean COH across all the CVs, as well as the standard deviation. The mean COH and its standard deviation are comparable across applications of different sizes.

Coupling

This is the degree of commonality of information across CVs. We define a *repetition count* of an IC as the number of occurrences of the IC in different CVs - 1. We define a metric called *Coupling* (COU) to measure coupling.

$$\text{COU} = \frac{\Sigma \text{repetition count}}{\Sigma \text{ IC}} \quad \text{where the summation is across ICs}$$

The COU metric ranges in value from 0 to infinity. A higher value indicates a higher coupling, which is less desirable. The COU metric is comparable across applications of different sizes.

Download Time

This metric is dependent on several endogenous variables not captured in CMU-WEB, as well as several exogenous variables listed in section 3, and is thus not measurable from a CMU-WEB schema.

Table 1: Summary information on the usability metrics derivable from CMU-WEB schema

Metric Name	Metric Range	Comparable Across Applications of Different Sizes?
LCRIC	0 to infinity	Mean and std. dev.
LCSTM	1 to infinity	Mean and std. dev.
GCHLWA	0 to infinity	Yes
GCCJ	0 to 2	Mean and std. dev.
COC	1 to infinity	Yes
COH	0 to 1	Mean and std. dev.
COU	0 to infinity	Yes

Usability of Completely Flexible Interfaces

The usability metrics for completely flexible interfaces are best tested empirically, since, to the best of our knowledge, so far there is no way to create a conceptual model that can model completely flexible interfaces.

Table 1 summarizes the seven metrics proposed in this work.

CONCLUSION AND FUTURE OPPORTUNITIES

In this work, we proposed a three dimensional classification of web applications. The classification provided insight into how the different design methodologies interact and can be used to create a WWW application. Specifically, developing a WWW application involves using design methods for each of the 3 dimensions in the classification.

While design methodologies already exist along both the *location of processing* and *the degree of support for interrelated events* dimensions, we identified a need for a conceptual model *along the structure of pages* dimension that facilitates design along this dimension. To fulfil this need, we first identified design questions that should be answered along this dimension, listing them as high level metrics. Next, we proposed CMU-WEB: a simple conceptual model that can be used in the analysis phase of a WWW application, as well as to reverse engineer an existing WWW application. We presented a list of seven low level metrics, whose values can be derived from CMU-WEB schema. This implies that web applications can be designed for better usability, by using CMU-WEB as the conceptual model along *the structure of pages* dimension. As in all analytic design methodologies, the advantage gained by using CMU-WEB for designing a proposed WWW

application is reduced time for implementation, and a better (more usable) application. We can also use CMU-WEB to compare existing WWW applications for usability, without doing any empirical testing. The approach is simple: create a CMU-WEB schema for each of the WWW applications, and derive the values of the seven metrics for each application.

We have created a CMU-WEB schema for one real life application so far. Our experience with it has been that it is reasonably simple to use, since it has very few components and semantic rules for consistency. Just as the critical issue in the ER model is the identification of entities and relationships, the critical issue in using CMU-WEB seems to be the identification of ICs and RICs.

CMU-WEB represents, to the best of our knowledge, a first effort to create a conceptual model for designing the usability of WWW applications (versus merely documenting the application). Our future research aims at using CMU-WEB for measuring the usability of more existing WWW applications, at identifying more metrics that can be measured using CMU-WEB, and at promoting the use of CMU-WEB by the academic and practitioner communities.

ENDNOTES

- ¹ Unless specified otherwise, clients in this paper mean “web browsers” and servers mean “web servers”.
- ² We ignore browser specific technologies like cookies that allow for maintenance of state between pages.
- ³ Examples include sound and image contained in a document.
- ⁴ Examples include protocol specific data such as GET requests from the client to the server, as well as alphanumeric data that is specific to the application.
- ⁵ In this work, we reference a large number of current technologies. Rather than referencing each one, we urge the reader to reference one of several excellent books on technologies of the WWW.
- ⁶ The display of XML documents is controlled by an accompanying XSL (extensible style sheet) which allows the same content to be displayed differently.
- ⁷ Many applications work around this by using proprietary technologies like cookies.
- ⁸ This can be presented in the application. Examples of ICs include: a textual paragraph describing an idea, an attribute (say, customer name) of a real world entity (say, customer), a .gif file; a .jpeg file and textual information

about a real world entity. Only text based information can be split into smaller ICs. A photograph, a movie or a piece of sound are considered to be single ICs. The problem of splitting these multimedia files into smaller ICs is for future research.

- ⁹ As an example of using client side processing to maintain state, consider an application that requires a complicated series of inputs from the user, where each input is dependent on the previous ones. A Java applet can take the user through these interrelated activities, and obtain the required input. Once these interrelated (input) events are done, it can then contact the WWW server with the sequence of inputs to perform a server side event.
- ¹⁰ We ignore processing that is part of the HTTP protocol.
- ¹¹ This, of course, is similar to notions of quality in relational database systems.
- ¹² The application is assumed to be running on a WWW server with a valid address.

REFERENCES

- Arnold, K., & Gosling, J. (1996). *The Java Programming Language*: Addison Wesley Pub. Co.
- Bajaj, A., & Krishnan, R. (1998, June). *Analyzing Models For Current World Wide Web Applications Using a Classification Space and Usability Metrics*. Paper presented at the Third Workshop on the Evaluation of Modeling Methods in Systems Analysis and Design, in conjunction with CAiSE.
- Berners-Lee, T., Caillau, R., Luotonen, A., Nielsen, H., & Secret, A. (1994). The World-Wide Web. *Communications of the ACM*, 37(8), 76-82.
- Bhargava, H. K., & Krishnan, R. (Forthcoming). The World Wide web: Opportunities for OR/MS. *INFORMS Journal On Computing*.
- Bichler, M., & Nusser, S. (1996). *Modular design of Complex web Applications with W3DT*. Paper presented at the Fifth Workshop on Enabling technologies: Infrastructure for Collaborative Enterprises: WET ICE, Stanford, CA, USA.
- Boar, B. H. (1992). *Implementing Client Server Computing*: McGraw Hill.
- Booch, G. (1994). *Object Oriented Analysis and Design with Applications*. Redwood City: Benjamin/Cummings.
- Botafogo, R. A., Rivlin, E., & Shneiderman, B. (1992). Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics. *ACM Transactions on Information Systems*, 10(2), 142-180.

- Bulman, D. (1991). *Refining Candidate Objects*. Computer Language, 8(1).
- Castellini, X. (1998). *Evaluation of Models defined with Charts of Concepts: Application to the UML Model*. Paper presented at the Third Workshop on the Evaluation of Modeling Methods in Systems Analysis and Design, in conjunction with CAiSE., Pisa, Italy.
- Chen, P. P. (1976). The Entity-Relationship Model: Towards a Unified Model of Data. *ACM Transactions on Database Systems*, 1(1), 9-36.
- Conklin, J. (1987). Hypertext: An Introduction and Survey. *IEEE Computer*, 20(9), 17-40.
- Deshpande, Y. L., Jenkins, R., & Taylor, S. (1996). *Use of Simulation to Test Client-Server Models*. Paper presented at the Proceedings of the Winter Simulation Conference.
- Drakopoulos, E. (1995). *Design Issues in Client/Server Systems*. Paper presented at the IEEE 14th Annual International Phoenix Conference on Computers and Communications.
- Garzotto, F., Mainetti, L., & Paolini, P. (1995). Hypermedia Design, Analysis and Evaluation Issues. *Communications of the ACM*, 38(8), 74-86.
- Isakowitz, T., Stohr, E. A., & Balasubramanian, P. (1995). RMM: A Methodology for Structured Hypermedia design. *Communications of the ACM*, 38(8), 34-44.
- Jefferey, R., & Stathis, J. (1996). Function point Sizing: Structure, Validity and Applicability. *Empirical Software Engineering*, 1, 11-30.
- Khare, R., & Rifkin, A. (1997). XML: A door to automated web applications. *IEEE Internet Computing*, 1(4), 78-87.
- Major, J. (1996). Client/Server capacity planning and hardware resource usage metrics, trends and relationships. *Capacity Management Review*, 24(2), 3-9.
- Martin, J., & Odell, J. (1992). *Object-Oriented Analysis and Design*. Englewood Cliffs, New Jersey: Prentice Hall.
- net.genesis, & Hall, D. (1995). *Build a Web Site*. Rocklin, California: Prima Publishing.
- Nielsen, J. (1993). *Usability Engineering*: Academic Press.
- Preece, J. (1994). *Human-Computer Interaction*. Reading, Mass.: Addison-Wesley.
- Schwabe, D., Rossi, G., & Barbosa, S. D.-J. (1996). *Systematic Hypermedia Application Design with OOHDM*. Paper presented at the Hypertext, Washington, D.C.
- Shneiderman, B. (1998). *Designing the User Interface*. (Third ed.): Addison Wesley Longman.

- Thuring, M., Hannemann, J., & Hake, J. M. (1995). Hypermedia and Cognition: Designing for Comprehension. *Communications of the ACM*, 38(8), 57-66.
- Troyer, O. M. F. D., & Leune, C. J. d. (1998, 1998). *WSDM: A User Centered Design Method for Web Sites*. Paper presented at the WWW7 Conference <<http://infolab.kub.nl/prj/wsdm/www7/final/paper.html>> accessed Feb. 20, 1999.
- Vora, P. (1997). Human Factors Methodology. In C. Forsythe, E. Grose, & J. Ratner (Eds.), *Human Factors and Web Development*: Lawrence Erlbaum Associates.

Chapter 13

The Effects of Using a Triangulation Approach of Evaluation Methodologies to Examine the Usability of a University Website

Dana H Smith, Zhensen Huang, Jennifer Preece, and Andrew Sears
University of Maryland, Baltimore County, USA

The objective of this study was to evaluate the current University of Maryland, Baltimore County website and identify problems that could be addressed in an upcoming re-design project. In meeting this objective, we used a combination of evaluation methods in order to triangulate and collect different perspectives on the problems. Heuristic evaluations were performed to gain an overview of the problems with the website. A total of fifty-four Information Systems students participated in this particular portion of the study. Next, focus group sessions were conducted to seek out what individuals want and need from the site, along with specific problems encountered. And finally, thirteen subjects performed usability testing to examine specific issues concerning navigation. Together, these methods provide three different but synergistic perspectives. By gathering test data, observing users, and interviewing a range of individuals on campus, we were able to collect a wide variety of information that was compiled, analyzed, and formally reported to the design group. The analysis of the data collected from the three techniques revealed several key issues in which expert recommendations were made for website redesign. But more importantly, the result of using a triangulation approach in this research illustrates the value of combining

inspection methods and testing to identify usability problems on a University website.

INTRODUCTION

The World Wide Web has become an important tool that Universities use to market their institution to prospective students and provide University information and services to the campus community. A previous study performed at the University of Maryland, Baltimore County (UMBC) reported that the web is the third most important source of information about a University, following a campus visit and a conversation with current students (Hearne, 1999). As users become experienced with computer and Internet usage, they become dependent on the quick and useful information that websites provide. Information is the central theme of any website. The more a site helps people find the information they are looking for, the more usable it is (Spool, 1999).

Figure 1: Original UMBC Homepage



Considering the growing importance of the Web as a tool, UMBC performs periodic evaluations in the form of online surveys that collect opinions and usage information from site users. Whether user information and behaviors are collected via online surveys or are provided to system designers at each stage of the design process, it is clear that the user must be part of the process to achieve a usable system (Goodwin, 1987). Additional feedback was obtained from a Website Effectiveness Study (Hearn, 1999), performed in 1999 by an external entity; which supplied the university with the opinions of prospective students.

To further the evaluation process, the university requested from the Department of Information Systems a usability study of the website in which results will be

considered in an upcoming redesign of the homepage (Figure 1). Based on usability problems and opportunities disclosed by empirical testing, one can produce a new version of the interface (Nielsen, 1993). This study will examine several usability inspection methods and user testing, but more importantly will explore the effects of combining several techniques in a triangulation approach in discovering interface usability problems. Several studies of usability inspection methods have discovered that many usability problems are overlooked by user testing, but that user testing also finds problems that are overlooked by inspection. This suggests that the best results are achieved by combining empirical tests and inspections (Nielsen, 1994).

Heuristic Evaluation

Heuristic evaluation is a usability engineering method in which a small set of evaluators examine the user interface to find possible problems using recognized usability principles (the “heuristics”) (Nielsen, 1992). In general, heuristic evaluation is best when performed by many evaluators, as one individual will never be able to find all the usability problems in an interface. It is also possible to improve the effectiveness of the method significantly by involving multiple evaluators. One advantage of heuristic evaluation is it is relatively inexpensive and provides quick results. Experts also say that over 75% of usability problems can be found using heuristic evaluations. However, one problem with this method is the fact that evaluators may not be the true users, and thus may overlook some usability problems.

The system usability principles that evaluators refer to are also called heuristic guidelines. Generally, typical guidelines can be:

- Use simple and natural dialogue,
- Speak the users’ language,
- Minimize user memory load,
- Be consistent,
- Provide clearly marked exits,
- Provide shortcuts,
- Provide good error messages, and
- Prevent errors. (Nielsen, 1992 Preece, 1993)

Focus Groups

Considered to be very much like well-designed meetings, a focus group session is a discussion-based interview that produces a particular type of qualitative data (Breakwell, 1995). The purpose of focus group sessions is to acquire the opinions and feelings of individuals on a particular topic. Used alone or in combination with other methods, the aim of focus groups is to get closer to participants’ understandings of and perspectives on certain issues (Breakwell, 1995). Again, accepting user input into the design process has many benefits and has become a

standard practice in systems development. Early focus on users and tasks is a recommended principle of usability design (Gould, 1985). The beauty in this method is its ability to explore a few people's judgments and feelings in great depth, and in doing so learn how end-users think and feel (Rubin, 1994).

Usability Testing

Usability testing is a method that employs techniques to collect empirical data while observing representative end-users using the product to perform representative tasks (Rubin, 1994). Usability testing is a very important tool that can reveal problems that designer/developers may not detect. The testing should include participants that represent real users that perform real tasks. Usability testing defines the acceptable performance of the system for specific types of users carrying out specific tasks (Preece, 1993). The goal of this method is to collect information about problems and use it to create a product that is easy to learn and use and that also provides functionality necessary for the user to accomplish the objective. If the site contains unclear or inconsistent information or is unpredictable and awkward to navigate, the user may have trouble completing each task. The usability test should reveal problems that a typical user could encounter.

Using these three techniques (heuristic evaluation, focus group sessions, usability testing) to evaluate the website will complement the former research that explored marketing aspects and user opinions, and attempts to discover the problems that current users encounter with the site. Alone, each can provide beneficial feedback, but together the results should reveal key issues that help identify valid problems. The purpose of this study is to examine the effects of a triangulation approach in evaluating the usability of a University website.

METHODS

The UMBC website is an extensive site, and it would be nearly impossible to design a usability test to detect every potential usability problem. Therefore, using the three evaluation methods previously mentioned in a triangulation approach might prove to be a more effective approach than any single method. The combination could also identify key issues requiring specific attention.

For the heuristic evaluation of the UMBC website, students from two sections of the Spring 2000 IFSM 303 course (Human Factors in Information Systems) were used as evaluators. Fifty-four students participated in the study. Heuristic guidelines from Nielsen and Preece were used to gather information. Using the guidelines and a worksheet, the students were asked to browse the UMBC web site with specific tasks in mind. While browsing, the students were asked to record on the worksheet any problems encountered and to identify the task and heuristic violation.

To complement the heuristic evaluations, focus group sessions were conducted as a second method of data collection. Using six to eight participants in each group, this study sought whether the information needs of various groups within the campus community were being met. Further, information regarding problems and experiences using the UMBC website was sought. As mentioned before, information is the central theme of any website. If the information a user requires is not available, then the site is not considered usable to the user.

During the planning stages of the project, it was decided that five different focus groups would be conducted, each consisting of six to eight subjects. Ideally, half the participants would be women; half would be inexperienced with the technology. Although there are numerous groups on campus to consider, this research specifically targeted the ideas, opinions, and experiences of undergraduate students (non-IFSM majors), graduate students (also non-IFSM majors), faculty, staff, and business/alumni. A campus-wide call for participation was issued via e-mail in late February 2000.

After developing and pilot testing the focus group questions, four focus group sessions were scheduled and conducted during March and April 2000. The groups ranged from two participants in the staff group to five in the graduate student group, and each session lasted about an hour. Most focus group researchers agree that between one and two hours is the standard duration for each session involving adults (Breakwell, 1995).

The moderator began the sessions with a brief introduction that explained the general purpose of the focus group and informed the participants that their comments would be considered anonymous and confidential. After personal introductions, the moderator presented three questions, one at a time, and allowed a 20-minute discussion for each. A colleague assisted the moderator by noting the general demographics of the group and comments to each question. Once each focus group session was complete, the participants performed a 30-minute usability test, which is explained next.

Usability testing was the third and final evaluation method used in this study. Usability experts recommend that 6 - 10 participants should be recruited to perform usability testing (Dumas, 1993). The usability test for this project consisted of eight tasks that were to be performed using the UMBC website. If carried out correctly, each task can produce UMBC information that is commonly used by many groups on campus. (e.g., search for e-mail addresses, course offerings, phone numbers, etc.). The focus group participants, who were undergraduates, graduate students, faculty, and staff, completed the usability test. Nearly all the tasks were simple or closed tasks where the user would find a specific answer somewhere on the website. One task was compound in nature, which required

the user to find information based on specific criteria. A final task required the user to browse the site and gather information about a specific topic.

Each task was followed by four questions. The first question requested the information that the user found, the second question asked the length of time needed to complete the task, and the third and fourth questions asked if any problems were encountered and what the problems were. This combination of questions can reveal valuable information on whether or not the information can be found, can it be found easily and in a timely manner, and whether any problems were encountered during the process.

RESULTS AND DISCUSSION

This section provides a brief overview of the results from each evaluation method, followed by a summary of key issues identified by using a triangulation approach.

Fifty-four students performed heuristic evaluations. As a result, 255 problems were reported. The results were analyzed, summarized, and the top seven are as follows:

- 29 complain of long pages,
- 25 mention use of color,
- 20 are related to broken links,
- 20 are related to navigation support,
- 18 suggest changing some of the menu names,
- 15 complain that they received outdated or incomplete information, and
- 15 report that they can not go back to previous pages, or can not find a link back to UMBC home page when they were in a specific sub-site.

Five focus group sessions were successfully conducted; one of which was the pilot study. Due to project deadlines, one targeted group (business/alumni) was not scheduled or conducted. In all, 19 subjects (15 women, 4 men) participated in the sessions. The recorded comments from each focus group session were analyzed and grouped according to theme, and an overview of the most common responses follow.

While the moderator sought both online information needs and problems encountered with the site, the overwhelming response across all groups was the need for more information. The groups not only wanted to find the information they need but also wanted to be informed of events occurring on campus. Very few problems with the site were discussed.

Naturally, there were many similarities between several of the groups. One issue that surfaced was the desire for information that already exists on the website.

Another suggestion from all groups was the need for information on how to use the website, or where and who to go to for various types of information on campus. Several groups also complained about outdated or non-existent web pages. The students desire and expect academic information on faculty web pages.

During the usability-testing phase, we found that while most of the users were familiar with the web site, there were still numerous problems encountered as the participants performed each task. On average, the length of time to complete the test was 20 minutes: the shortest completion time was 15 minutes, the longest was 35 minutes. Although there were a variety of problems overall, 37% of the users had problems with each task. The majority of the problems encountered centered on the fact that the users were unsure where to find the information; therefore they had trouble completing the tasks in a reasonable amount of time.

Thus far, a summary of the major outcomes of each evaluation method has been presented, and independently they provide insight into potential problems with the site or deficiencies of information. While each method provides helpful feedback, analyzing together the results from all three methods helps to identify key issues that were summarized and reported to the design team.

The key issues are organized into three categories: Navigation Problems, Information Design Issues, and Information Needs. Finally, a listing of “other requests” is presented which describe numerous ideas that the study participants offered (primarily through the focus group sessions).

Navigation Problems

Like most aspects of usability, navigation is invisible when it is working. But when there is a problem, users can get completely stuck. In fact, navigation problems frequently caused users to give up (Spool, 1999). There are six general guidelines suggested that encourage good navigation: avoid (1) frames, (2) orphan pages and (3) long pages; provide (4) navigation support and (5) consistent look and feel; and (6) avoid narrow, deep, hierarchical menus (Preece, 2000). Although all are not discussed here, some are mentioned to resolve several of the findings in this research.

One issue that repeatedly generated discussion during the focus group sessions was the lack of information on how to use the website and where to find things online. The heuristic evaluations and usability testing verified that many users experience problems while navigating the site or searching for specific information. During usability testing, numerous participants did not know where to begin when presented with a specific task. Providing good navigation support by adding a *site map* link on the home page may increase user understanding of how the site is organized, thereby decreasing the confusion of not knowing where to go to find things online. Another guideline to consider for this particular problem

would be to provide a consistent navigation panel on each page. By providing consistency, users feel that they know where they are at all times and not lost in the site. It has also been suggested that people learn faster when what they see and do is consistent (Schneiderman, 1992). Another important feature that the home page should offer is a prominent *search* feature, as many users are search-dominant and do not want to bother navigating to their destination link-by-link (Nielsen, 1999). Lastly, providing descriptive link titles may increase user understanding and usability of the site.

Social event information, both on and off campus, is an important issue for students. This particular topic was discussed at great length during all three of the student focus group sessions. Many mentioned that they learned of event information by reading the posters on the walls in the student center stairwell or heard about an event after it happened. It was obvious during the focus group sessions that the students did not know that this information is available online. Similarly, the usability test revealed that some students did not know where to find off-campus activities on the website. Again, a strong site map would help with this particular issue. But, because this topic seems to be very important to the students, providing additional descriptive information on the homepage regarding social events or activities may improve their awareness of the existing information.

One of the top five problems identified during the heuristic evaluation involved menu names. Many of the students found the names confusing or unclear. The results of the usability test revealed similar problems, as some of the students were unsure where to look for information because the link titles were not descriptive. The better users could predict where a link would lead, the more successful they were in finding information (Spool, 1999). Once again, for navigation purposes, provide clear understandable menu and link names.

The top category of problems found during the heuristic evaluation was the occurrence of long web pages. Students complained about scrolling to find information. Similar complaints were reported on the usability test. Web page designers should avoid long pages that force scrolling (Preece, 2000). Users do not like to read material on the screen. They prefer to scan, and fail to scroll to the bottom of long pages (Nielsen, 1998). Using hypertext to split up long information into multiple pages (Nielsen, 1999) is one way to outline and organize information when attempting to design less lengthy pages.

Another important result of both the heuristic evaluation and usability test is a problem with broken links. This can be very frustrating to the user when searching for online information. Countless and consistent broken links can lead to dissatisfied users, thus a bad reputation. Maintainers of the website should continuously monitor for broken links and repair them as needed.

Information Design Issues

There are several guidelines that support good information design: avoiding (1) outdated or incomplete information, (2) excessive use of color, and (3) gratuitous use of graphics and animation; and providing (4) good graphical design and (5) consistency (Preece, 2000). Two of these guidelines were addressed during the heuristic evaluation and focus groups. *Outdated and incomplete information* generated a fair amount of discussion, but because it also addresses the information needs of the users, it will be discussed in the next section. *Excessive use of color* was the other issue that some of the participants, mostly students, criticized.

The heuristic evaluations and focus group sessions revealed a problem with color choice and consistency. Several students commented that some of the color choices decreased readability; some reported inconsistencies between pages, while others commented on the use of red. Using soft background colors with contrasting color for text should improve readability. Optimal legibility requires black text on white background, and big enough fonts that people can read (Nielsen, 1999). As a general rule, color is useful for indicating different kinds of information, and a change of color should signal a change in information type (Preece, 2000). There are also general color standards that apply to the web and should be considered here. One such standard is the use of blue as a link color, and links that have been viewed usually change to purple or red in color. Using standard link colors provides consistency whether in the UMBC site or elsewhere on the web. Finally, carefully check the use of the color red. Text this color is hard to read, and in some cultures the color red signifies danger.

Information Needs

While seeking the experiences, opinions, and needs of participants during the focus group sessions, we found that there were two features of the site that many of the participants found useful. Although they suggested adding a few more options, most agreed that the capabilities of myUMBC are far better than past systems. The students mentioned that adding student library account information to myUMBC would be helpful. Another feature of the site that was mentioned in a positive manner was the library homepage. Although a few students found it to be cluttered, and many suggested that more e-journals should be available, overall they found the library site to be a good online resource.

The leading complaint in the area of information needs was outdated and incomplete information. All five focus groups identified this as a problem, as did a number of heuristic evaluators. The participants mentioned dissatisfaction with non-existent and out-of-date faculty, department, and sports web pages. Some students mentioned the need to (1) view online course syllabi, (2) seek a way to contact part-time faculty, and (3) research departmental information. Outdated



or incomplete information should be avoided as it creates a poor impression with users (Preece, 2000). Again, closely monitoring the information provided on the UMBC website and keeping it up to date will provide users with confidence in the information that they received.

Finally, many suggestions were offered regarding the need for additional online information. Once again, usability testing and heuristic evaluations would not have revealed the lack of some useful online information as did the focus groups sessions.

Examples:

- Add student library account information to myUMBC.
- For advisement purposes, faculty expressed the need for online student academic information.
- Students mentioned the need for more e-journals.
- Faculty requested a way to check library reserves online and easy access to library catalogue system.
- Online grant writing support.
- Online reservations for computer labs and AV equipment.
- Online status information from bookstore on book orders.

CONCLUSION

Using a combination of evaluation methodologies was beneficial in this website review. First, additional problems were identified when using more than one approach. Conducting focus group sessions identified various online information needs that otherwise would not have been discovered using heuristic evaluations

or usability testing. As mentioned before, information is the central theme of any website. The more a site helps people find the information they are looking for, the more usable it is (Spool, 1999). To further this study, the results of the focus group session could be followed up with another online survey that is more specific to the topic of online information needs. This evaluation tool could possibly validate the results of the focus group sessions.

Second, when using more than one evaluation technique, specific problems can be validated when the problem is identified by more than one technique. It was important to website management team to have valid recommendations to use for the redesign project. The expert recommendations offered to the UMBC website management team were based on problems that were identified by more than one technique. Figure 2 displays the new UMBC homepage that was launched in August 2000.

While this study focused on heuristic evaluations, focus groups, and usability testing, it would be interesting to build on this study to include other inspection methods. Perhaps including cognitive or heuristic walkthroughs would result in different outcomes. But overall, examining the effects of a triangulation approach could prove to be beneficial in website evaluations.

ACKNOWLEDGEMENTS

The authors want to thank Mr. John Fritz for his assistance throughout the project. We also want to thank all the subjects for their participation in the study.

REFERENCES

- Breakwell, G., Hammond, S., & Fyfe-Schaw, C. (Eds.). (1995). *Research Methods in Psychology*. London : Sage Publications.
- Dumas, J. S. & Redish, J. C. (1993). *A Practical Guide to Usability Testing*. Norwood, NJ: Ablex Publishing Co.
- Goodwin, N. (1987). Functionality and Usability. *Communications of the ACM*, 30(3), 229-233.
- Gould, J. D. & Lewis, C. (1985). Designing for usability: Key principles and what designers think. *Communications of the ACM*, 28(3), 300-311.
- Lipman Hearne, (1999). *UMBC Web Site Effectiveness Study*. [Online]. Available: <http://www.wses.com/wses/g.html> [2000, March 22].
- Nielsen, J. (1992, May 3 - 7) Finding usability problems through heuristic evaluation. Paper presented at the Proceedings on the Conference on Human Factors and Computing Systems: CHI'92, Monterey, CA.
- Nielsen, J. (1993). *Usability Engineering*. San Diego, CA: Academic Press.

- Nielsen, J. (1998) *About Jacob Nielsen*, [Online]. Available <http://www.useit.com> [2000, April 20].
- Nielsen, J. (1999). *Designing Web Usability*. Indianapolis, Indiana: New Riders Publishing.
- Nielsen, J. & Mack, R.L.(1994). *Usability Inspection Methods*. New York, NY: John Wiley & Sons, Inc.
- Preece, J. (1993). *A guide to Usability: Human Factors in Computing*. Reading, MA: Addison-Wesley.
- Preece, J. (2000). *Online Communities: Designing Usability and Supporting Sociability*. New York, NY: John Wiley and sons.
- Rubin, J. (1994). *Handbook of Usability Testing*. New York, NY: John Wiley and sons.
- Schneiderman, B.(1992). *Designing the user interface: Strategies for effective human-computer interaction* (2nd Ed) Reading, MA: Addison-Wesley.
- Spool, J. (1999). *Web Site Usability: A Designer's Guide*. San Francisco, CA: Morgan Kaufmann Publishers, Inc.

Chapter 14

Adaptive Web Representation

Arno Scharl

Vienna University of Economics, Austria

Users tend to have varying preferences regarding multimodal access representations. The number of alternatives provided by paper-based media is inherently limited. Adaptive hypertext applications do not share this limitation. This paper classifies them into three categories of information, and their corresponding interface representation: Content of documents, primary navigational system comprising links between and within these documents, and supplemental navigational systems such as index pages, trails, or guided tours.

INTRODUCTION

This chapter introduces a classification framework for adaptive Web representations. Adaptive components extend an information system's functionality and replace general-purpose documents that are written according to a wide audience model and the user's anticipated needs. While being motivated by a user-centered design perspective, the question goes beyond the scope of interface design or document presentation. Current efforts include the development of Web-based architectures that take advantage of adaptive system behavior. Emphasizing the role of annotations, the described framework comprises three categories of information and their corresponding interface representation: Content of documents, primary navigational system including the links between and within these documents, and supplemental navigational systems such as index pages, trails, guided tours, or overview maps.

ANNOTATIONS

Most scholarly articles and books exemplify explicit hypertextuality in non-electronic form by using a sequence of numeric symbols to denote the presence of footnotes, signaling the existence and location of subsidiary texts to the main document (Burbules & Callister, 1996; Snyder, 1996). However, as far as printed material is concerned, the reader rarely is exclusively attracted by the footnotes and “becomes fascinated with the nonlinearity and incompleteness of such a collection of fragments, just as one does not give up a novel to start reading the phone directory” (Rosello, 1994). Annotations are similar to the concept of a footnote in traditional texts, but usually are added by an author or, collaboratively, by a group of authors different from the producer of the main document. Common interface representations of annotations include textual additions or various visual cues such as icons, highlighting, or color coding.

CONTENT-LEVEL ADAPTATION

Technically, the term content-level adaptation refers to all different forms of data embedded in hypertext documents. In practical terms, however, almost all prototypes and implemented systems concentrate on textual segments, neglecting visual and audiovisual forms of data. A good indicator for the granularity of adaptivity is the average length of textual segments. Scope for contextual variability is introduced by establishing an independent format for storing these segments, and by incorporating a flexible mapping algorithm to provide various types of traversal functions. A large number of very short textual elements ensure maximum flexibility and (potentially) a very exact match between the presented information and the user’s actual needs. However, the required efforts to maintain the database as well as the rule set significantly increase with the number of distinct elements.

Nielsen (1999) classifies content-based adaptation into the following categories: *Aggregation* (showing a single unit that represents a collection of smaller ones), *summarization* (representing a large amount of data by a smaller amount; e.g., textual excerpts, thumbnails, or sample audio files), *filtering* (eliminating irrelevant information), and *elision* (using only a few examples for representing numerous comparable objects). One of the simpler but nevertheless quite effective low-level techniques for content adaptation is conditional text (also referred to as *canning* or *conditionalization*), which requires the information to be divided into several chunks of texts (Brusilovsky, 1998; Knott, Mellish, Oberlander, & O’Donnel, 1996). Each chunk is associated with a condition referring to individual user knowledge as represented in the user model. Only those chunks appropriate for the user’s current level of domain knowledge are considered when generating the document. The granularity of this technique can range from node-

level adaptivity (i.e., storing different variations of whole documents) to very fine-grained approaches based on sentences or even smaller linguistic units.

The term *stretchtext* denotes a higher-level technique. The idea is to present a requested page with all stretchtext extensions that are non-relevant to a particular user being collapsed. While reading the document, the user is able to collapse optional chunks of text and uncollapse the corresponding terms whenever s(he) desires. Applications of stretchtext can be categorized along two dimensions (Boyle, 1998): *Placement* of the text relative to the original, either at the beginning or the end, embedded inside the old lexia or completely replacing it; and *granularity*, understood as the average length of lexias, usually based on graphical forms such as words, sentences, or paragraphs. By activating and closing stretchtext extensions, the user creates summary and ellipsis, by means of which the articulated discourse is shortened (Liestøl, 1994). Consequently, one of the main advantages of stretchtext is that it lets both the user and the system adapt the content of documents, giving the user the possibility to “override” the information stored in the user model.

The most powerful content adaptation technique is based on frames and presentation rules where slots of a frame can contain several different explanations of the concept, links to other frames, or additional examples. Usually a subset of slots is presented in order of decreasing priority. Research on natural language generation, which aims to produce coherent natural language text from an underlying representation of knowledge (Milosavljevic & Oberlander, 1998), provides valuable insights for implementing such an advanced content adaptation technique.

LINK-LEVEL ADAPTATION

Basic link-level adaptation techniques, which usually aim at decreasing the cognitive overload caused by complex Web information systems, can be grouped into four categories:

- Providing relevant starting points in the information space (Vassileva, 1998).
- Influencing link perception: *Hiding* actually restricts browsing to smaller subspaces for inexperienced users. *Dimming* decreases the cognitive overload as well, but leaves dimmed links still visible - and traversable, if required. *Highlighting* attributes include boxing, blinking, color, texture, and reverse video. As lateral and temporal masking negatively impact the other attributes, color coding remains the method of choice for most applications).
- Sorting: Presenting recommendations in the form of a sorted list of links transforms the complex problem of advice-giving into the much simpler problem of rank ordering a list (Kaplan, Fenwick, & Chen, 1998).
- Annotating via semantic link labels (history-based versus user model based; see above),

Due to limitations in the current infrastructure of the World Wide Web, most systems lack mechanisms to show the mutual dependencies and co-constitution among possible categories of thought (Kolb, 1994). Many systems do not provide links that allow the user to anticipate where the link will lead them, or to clearly distinguish them from other links located in the vicinity. Semantic link labels, which can be regarded as a subcategory of annotations from a theoretical perspective, address this problem of inexpressiveness (Mayfield, 1997) by conveying information about a link's purpose and destination, its relevance in the current context, its creator, or its date of creation. By helping users evaluate whether to follow a link without having to select it, link labels increase cohesion and help maintain context and orientation in non-linear presentations.

Current Web browsers only support history-based signaling whether a link has already been followed by the user. This is only a very basic mechanism, not tapping the full potential of hypertextual navigation. It can easily be extended by including a categorization of hypertext links. Labeling them in a consistent manner according to their type comprises the following categories: Intratextual links within a document (related content marked by anchors, annotations, footnotes, citations, figures, etc.), intertextual links between documents, and interorganizational links to systems of other corporations.

Combining these categories with explicit annotations (e.g., numerically or via color coding) of the percentage of people who followed each of the links off the current page further increases the system's usability. This mechanism may also incorporate an adaptive component, considering only those links relevant to a user and computing percentages exclusively for this subset.

META-LEVEL ADAPTATION

Powell et al. distinguish three types of navigational support: textual, visual, and metaphorical (Powell, Jones, & Cutts, 1998). Covering all three categories, this section will focus on so-called "*supplemental navigation systems*" (in contrast to the primary navigational system comprising contextual and non-contextual links as discussed in the previous section). Supplemental navigation systems are used to locate and interpret a given item of information, providing full context by verifying the relation between different items and the virtual spaces surrounding them. They should include mechanisms to signify the user's current location, and to retrace her individual steps.

Such systems, however, become more than mechanisms to navigate a virtual space; they become crucial textual elements themselves, replete with their own interpretive assumptions, emphases, and omissions (Burbules & Callister, 1996). To support the different preferences, levels of technical knowledge, and cognitive

styles of their users, advanced Web applications usually employ a combination of the following supplemental navigation systems:

- *Site maps* (local and global);
- Site indexes and *tables of contents* add a second level structure (often alternatively referred to as *thesaurus*, *semantic net*, *domain knowledge*, or *index space*) on top of the basic document hypertext structure (Mathé & Chen, 1998).
- *Direct guidance* and retrospective access via chronological or parameterized *backtracking* to relate the current context to previously covered information.
- *Hierarchical bookmark lists* that enable users to tag elements perceived to be of long-term importance so that they can directly return to a particular document without having to remember and retrace the original pathway (Burbules & Callister, 1996).
- Access to *search engines* and general *databases* using queries, eliminating the need to pre-organize the information space in a hierarchical way.

REFERENCES

- Boyle, C. (1998). Metadoc: An Adaptive Hypertext Reading System. In P. Brusilovsky, A. Kobsa, & J. Vassileva (Eds.), *Adaptive Hypertext and Hypermedia* (pp. 71-89). Dordrecht: Kluwer Academic Publishers.
- Brusilovsky, P. (1998). Methods and Techniques of Adaptive Hypermedia. In P. Brusilovsky, A. Kobsa, & J. Vassileva (Eds.), *Adaptive Hypertext and Hypermedia* (pp. 1-43). Dordrecht: Kluwer Academic Publishers.
- Burbules, N. C., & Callister, T. A. (1996). Knowledge at the Crossroads: Some Alternative Futures of Hypertext Learning Environments. *Educational Theory*, 5(4).
- Kaplan, C., Fenwick, J., & Chen, J. (1998). Adaptive Hypertext Navigation Based on User Goals and Context. In P. Brusilovsky, A. Kobsa, & J. Vassileva (Eds.), *Adaptive Hypertext and Hypermedia* (pp. 45-69). Dordrecht: Kluwer Academic Publishers.
- Knott, A., Mellish, C., Oberlander, J., & O'Donnell, M. (1996). Sources of Flexibility in Dynamic Hypertext Generation. Paper presented at the *8th International Workshop on Natural Language Generation*, Herstmonceux Castle, UK.
- Kolb, D. (1994). Socrates in the Labyrinth. In G. P. Landow (Ed.), *Hyper/Text/Theory* (pp. 323-344). Baltimore: Johns Hopkins University Press.
- Liestøl, G. (1994). Nonlinearity and Literary Theory. In G. P. Landow (Ed.), *Hyper/Text/Theory* (pp. 87-120). Baltimore: Johns Hopkins University Press.
- Mathé, N., & Chen, J. R. (1998). User-Centered Indexing for Adaptive Information Access. In P. Brusilovsky, A. Kobsa, & J. Vassileva (Eds.), *Adaptive Hypertext and Hypermedia* (pp. 171-207). Dordrecht: Kluwer Academic Publishers.

- Mayfield, J. (1997). Two-Level Models of Hypertext. In C. Nicholas & J. Mayfield (Eds.), *Intelligent Hypertext: Advanced Techniques for the World Wide Web* (Vol. 1326, pp. 90-108). Berlin: Springer.
- Milosavljevic, M., & Oberlander, J. (1998). Dynamic Hypertext Catalogues: Helping Users to Help Themselves. Paper presented at the *9th ACM Conference on Hypertext and Hypermedia (HT-98)*, Pittsburgh, USA.
- Nielsen, J. (1999). User Interface Directions for the Web. *Communications of the ACM*, 42(1), 65-72.
- Powell, T. A., Jones, D. L., & Cutts, D. C. (1998). *Web Site Engineering: Beyond Web Page Design*. Upper Saddle River: Prentice Hall.
- Rosello, M. (1994). The Screener's Maps: Michel de Certeau's "Wandersmänner" and Paul Auster's Hypertextual Detective. In G. P. Landow (Ed.), *Hyper/Text/Theory* (pp. 121-158). Baltimore: Johns Hopkins University Press.
- Snyder, I. (1996). *Hypertext: The Electronic Labyrinth*. Melbourne: Melbourne University Press.
- Vassileva, J. (1998). A Task-Centered Approach for User Modeling in a Hypermedia Office Documentation System. In P. Brusilovsky, A. Kobsa, & J. Vassileva (Eds.), *Adaptive Hypertext and Hypermedia* (pp. 209-247). Dordrecht: Kluwer Academic Publishers.

Chapter 15

Usability: Changes in the Field A Look at the System Quality Aspect of Changing Usability Practices

Leigh Ellen Potter
Griffith University, Queensland

System Usability is becoming increasingly important in situations where assistance in the operation of a system is not readily available for the user. Traditionally, usability measures have consisted of post development testing in a usability lab. Usability practitioners are recognising the need for innovative procedures to incorporate usability in system development at an earlier stage than traditional testing allows and User Centred Design is an approach that meets this need. The objective of this chapter is to examine traditional usability testing and compare it to user centred design practices focusing on the resultant quality of the information system. An examination of literature concerning the two approaches is presented and comparisons made to a case study of a large Australian organisation utilising both measures. The experiences of developers and users within the organisation are presented, and the perceived quality of systems developed using both approaches is examined.

INTRODUCTION

The quest for a quintessential definition for quality continues to challenge many researchers. Quality in Information Systems can be viewed from multiple perspectives. According to Eriksson and Torn (1991), from a technical perspective it can focus on

efficiency of systems and processing. From a business point of view, it can focus on an increase in profitability. From the users point of view, it can focus on increased ease of use in a system and support of their work practices. ISO 8402 (1994 - Quality Management and Quality Assurance) describes quality as the totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs.

The objective of this chapter is to examine traditional usability testing and compare it to user centred design practices focusing on the resultant information system quality. In a comprehensive review of information systems literature, DeLone and McLean (1992) present such features as flexibility, usefulness and reliability as indicative of system quality. Many features that enhance the quality of a system are intangible and difficult to describe and measure, however it is these intangibles that draw the difference between the information quality of a system and traditional ideas of software quality. Many strategies have been developed to address these factors including a focus on system usability. Usability is defined in ISO 9241-11 (1998 - Guidance on Usability) as the “effectiveness, efficiency, and satisfaction with which specified users can achieve specified goals in particular environments.” Traditionally, usability practices have involved testing in laboratory situations. As the area has developed, the need for new usability practices has arisen and techniques such as field studies, workshops, prototyping, and user centred design have emerged.

Information System Quality

In addressing information system quality in this context, I refer to the aspects of process improvement and use improvement. Process improvement, or process quality, refers to the view that improvements in a production process improve the quality of the final product, and the product relies on these processes. In examining usability practices, I will be discussing different approaches to the process of system development and the resultant quality of the system.

Use improvement from a quality point of view refers to the aspects of a system that make it more effective for use. Myers et al. (1997) describe an effective information system function as one which provides the appropriate information to assist users in performing their work. This is a direct reflection of the aims of usability practices as defined in ISO 9241-11. Reijonen and Kesti (1994) describe the usability of information systems as an indicator of system quality and state that the evaluation of the usability of a system has assisted in improving system quality at the interface level where users interact and obtain information.

Examining information system quality from a usability perspective involves a user-based view of quality. This is not to exclude other quality measures in a different situation. In this discussion, when referring to a system user, I am referring to the person who interacts with the system on a regular basis in the performance of their work. This

definition is independent of the organisational status of the user. When examining a system a user considers its performance and behaviour in addition to technical properties, such as accuracy and timeliness (von Hellens, 1997). Targeting user based quality is fraught with difficulty. "This is an idiosyncratic and personal view of quality, and one that is highly subjective (Garvin, 1984, p27)." System attributes that can be examined for information system quality from this perspective include ease of use, ease of learning, flexibility in use, and security (Salmela, 1997). The assessment of these attributes is largely subjective, however Salmela also states that poor use quality will lead to an increase in system costs, particularly in learning and using the system.

Usability

The user's point of view is central to the usability field. Ovaska draws the relationship between task, system, goal and user and describes usability as "the totality of all factors that enhance or make possible the user's goal attainment" (Ovaska, 1991, p48). Flexibility, usefulness, reliability, effectiveness, efficiency and user satisfaction can be added to the user based quality attributes identified as attributes indicative of usability. Early attempts to evaluate and improve the quality of these attributes consisted of usability testing in labs set up for that purpose. These labs went some way towards demonstrating the deficiencies of a system, however the testing was frequently performed too late in the development cycle to enable significant changes to be made to the system. In more recent years, efforts have been made to move away from the usability lab or at least to change the nature of the testing. Tools were developed to enable testing to be performed earlier in the development cycle, and the importance of user participation in the process was emphasised. A new approach emerged – user centred design.

Usability Testing

Usability engineering and testing emerged in the mid-1980's and testing was viewed as an alternative to software quality assurance testing (Borgholm and Madsen, 1999). Usability testing labs were designed to resemble the work place and consisted of separate rooms divided by one way mirrors with observers behind the glass and audio and video equipment recording the users actions. The user was on their own in the lab and encouraged to think aloud as they worked their way through tasks or scenarios designed to test particular system features. The lab is limited in how well it can replicate the users normal working environment. Labs are still used (The May 1999 issue of Communications of the ACM described usability approaches undertaken by six companies and five described their use of usability labs as high), however the method of their use is evolving and alternatives are being developed. Iterative testing is becoming more common, and the involvement of a facilitator in the lab with the user is increasing. Early testing of the system design is emerging, where testing concentrates on overall

design and system concepts. Gardner (1999) feels that lab testing should fill a secondary role in usability practices.

User Centred Design

User Centred Design (UCD) is the process focusing on the daily life and experience of the user whereby users are actively involved in system design from an early stage of development. This extends the involvement of the user in requirements discussion into system design, and includes an analysis of current work practices and the impact the new system may have on these and it will involve both the users and the system designers (Wilson et al., 1997). Paper prototyping is a common tool in UCD allowing user evaluation of design options. User input to the design process is crucial.

This style of user participation overcomes many obstacles in system development. The use of dialogue between the users and developers demonstrates user needs and practices not otherwise viewed and allows an expansion from an evaluation of a product to an active involvement investigating new design possibilities. Engaging the developers in dialogue with the users allows the transferal of these possibilities (Buur and Bagger, 1999).

A challenge in UCD that is widely discussed by usability professionals is attracting the appropriate participants for the design process. The process of garnering the support of the appropriate participants and proceeding with an interactive design has been described as time-consuming, and the practice of UCD as a labour-intensive method of design (Vredenburg, 1999).

METHODOLOGY

This paper presents an examination of literature concerning the testing approach and the UCD approach and explores and compares the impact each is purported to have on the quality of the completed information system. This is achieved by examining the effectiveness and efficiency of the system in supporting the users in the completion of their work. A case study of a large Australian organisation will be presented where systems have been developed using each of the described approaches. Fifteen people were interviewed and a further nine were questioned as part of a larger study in usability undertaken as part of post graduate research (Potter, 1999). Where appropriate in discussion, the participants are named according to their primary role in the organisation, ie user1, developer1 and so on. The systems and participant experiences will be examined from an interpretive viewpoint using information from interviews with system users, developers, and executives and as such represents the subjective experience of one organisation. The results will be compared to findings from literature.

A Case Study

Research has taken place within a large, established Australian organisation (Potter, 1999). The organisation has offices in the capital city of each state. The procedures

in place within the organisation are standardised across the different offices and are traditional in nature. A hierarchy exists with a dominant Role culture as described by Handy (1985) that drives the organisation, with many departments operating within a Task culture. The organisation strives to be an international leader in its field, and to this end established a usability department in the head office in 1996. This has been a pilot exercise, with much of the work completed centrally. The department was officially launched mid 1999. The goal of the department is to introduce usability practices to the organisation and it achieves this through workshops in UCD and usability testing of systems in the usability lab in the head office.

The organisation has always emphasised the importance of the user in gathering system requirements. In the past, this process has not followed a set structure, and developers have described situations where the development teams have made incorrect assumptions concerning user needs. One executive described developers as designing systems around their own reality, rather than around the reality of “the people who in the end are going to have to use it to do their job (EXECUTIVE1).” This has led to the development of systems that are poorly utilised. Through the usability approach, user needs are identified and addressed in development.

FINDINGS

In literature, a demand exists for an earlier initialisation of usability practices. Traditional usability testing late in the development process does not allow for major changes, and usability professionals will be of increasing benefit if involved in development projects from the beginning (Gardner, 1999). More usability activities can be performed in the initial stages of the development cycle than can be performed later in development (Muller and Czersinski, 1999). Alternative approaches to lab testing are made possible by prototyping tools that allow early system evaluation (Dolan and Dumas, 1999) and these are utilised in UCD.

Many developers are interested in an involvement in different stages of development. Involvement throughout the process fosters greater collaboration (Borgholm and Madsen, 1999). This involvement includes a greater use of paper prototypes in early system development, rather than computer based prototypes, representing a move away from the technology environment towards the users environment. This process allows the central focus to be on the users everyday experiences (Gardner, 1999). “It is no longer regarded as sufficient to have the users participate as testers in fairly controlled lab environments. (Borgholm and Madsen, 1999, p96)”

Usability Testing

The organisation currently uses one usability lab situated in the head office. This means that at present development occurring outside the head office faces logistical problems and added expense in utilising the facility. This limits the scope of use for the

lab, and introduces a division within the organisation between the 'haves and have-nots'. Suggestions have been made to implement videoconferencing facilities to allow remote viewing. To date, this has not been initiated.

An advantage for traditional testing identified by developers in this organisation is that it is a tangible demonstration of the importance of usability. The developers are given the opportunity to see users struggling with the systems they have designed. It is described as a method of conversion, even for managers in many cases. In many ways testing is seen to have the least impact on developers, as they are not required to change their development approach as dramatically as is required for UCD.

There are a number of projects and developers within the organisation currently utilising the traditional late testing approach to usability. The experiences and attitudes of developers and users towards testing will be represented here by the experiences of a particular developer working on a corporate system, and a job tracking system, as these findings are indicative of the general trend for users and developers within the organisation.

DEVELOPER1 is dedicated to usability and is committed to testing, however cost is a concern for her. She is a self confessed developer of "some pretty unusable systems." DEVELOPER1 is in charge of a development team responsible for developing a corporate system that was tested on completion in the usability lab. It failed the usability testing. She stated that this was a remarkable learning experience for her, and opened her eyes to the importance of usability. As a result of the testing, the system was redeveloped and the usability of the system was improved. The benefits of the process included an improved awareness of the role of the user in system development on the part of the development team and their immediate superiors. The final system took longer to complete, and whilst more usable, did not incorporate the full range of requirements due to the lateness of error discovery. The system quality was improved by incorporating a higher degree of efficiency and reliability, however the use quality was not maximised as a number of user requirements were not addressed. The system was seen as less effective by the users because of this.

An example of a job tracking system developed without UCD was examined. The system is a core system across the entire organisation. It was tested upon completion, and was described by one executive as being poorly received, convoluted and failing to meet 90% of the required business needs. It failed to provide the required level of flexibility, ease of use and ease of learning and was seen to be inefficient and ineffective. The system required a complete rebuild. This process is now underway using UCD under the instigation of the system designer. Management and users describe a level of satisfaction with the modules of the system that have been completed in this manner.

Both the system users and developers feel that a drawback to traditional usability testing is the late point in the development cycle that the testing occurs in this organisation, reflecting the findings reported in literature. System users express the desire to be

involved earlier in the development process, stating that testing does not capture their requirements. It is felt that the alterations that can be made to the system at this stage are minimal. A more innovative approach to testing is the partial testing of early prototypes and this is felt to be more helpful in producing beneficial changes to the system and improving the quality of the final product.

User Centred Design

Developers who have experienced usability testing report a new appreciation and dedication to usability. This was also reported to occur in UCD. Developers describe it as a slower conversion, but no less valid or strong. Developers and users dedicated to UCD seem to see the link between an initial outlay in development rewarded by a later return in an improved system more clearly than testers, and this appears to be a factor in their preference for UCD.

User reaction to UCD is described by developers, executives, and participating users as very positive. It is felt by users and developers to improve user productivity and to improve developer satisfaction with system. Several users felt that UCD improved the timing of the data – “it’s quicker, at your fingertips (USER2),” and it contributes to making data entry easier and more accurate as the user has provided an indication of what information is required and how the screen should present the input request. This is then felt to improve productivity and lead to time benefits and use improvement.

The nature of UCD contributes to a greater consumption of resources at the beginning of the development life cycle. One executive stated that 30 – 40% of development costs occur at the beginning of the development. The process is time consuming, particularly in the early stages of development, however developers feel there are greater benefits in the long term, including less overall time spent in development and early identification of errors, which is a benefit acknowledged by all developers interviewed. The early identification of errors and weaknesses possible with the utilisation of UCD brings significant quality benefits for the final system. “Doing it early means you come up with a much better idea of the scope of what you need to do (DEVELOPER2).” It minimises system rewrites at the end of development and reduces the costs associated with this practice improving the effectiveness and efficiency of the system.

With the introduction of usability measures has come a push for internal standards to guide system development. At present, there is no formal quality department within the organisation to maintain quality standards and prior to the introduction of the usability department, there had been no universal set of development standards for the organisation. This situation has improved with the introduction of UCD, as the approach consists of prescribed steps and guidelines. Several developers who have undertaken UCD have drawn up guidelines or standards for devel-

opment within their own departments and have made these available to the general developer community within the organisation. These tend to be application specific, and build on the UCD guidelines. They are seen to contribute to greater consistency of development within the organisation, improving reliability.

Systems built using UCD are reported to assist the handling of information by improving data quality and ease of use. One user described this according to work-arounds. He stated that if a system was developed according to user specifications such as is achieved with UCD, it would be used in the appropriate fashion. He described systems within the organisation developed without UCD where usability was low and the user was inclined to attempt to circumvent the system by trying different solutions. He described this latter incident as time consuming and error prone.

Involvement is felt to improve efficiency and effectiveness (USER2) and ease of access (USER3). DEVELOPER3 stated that UCD improved data quality "because the software becomes easier to use and they (the users) can do a lot more of the editing themselves." Designers and executives describe UCD as an excellent methodology for dealing with complexity by clarifying user needs and business practices and introducing a formalism to the development approach. It is described as a common language between users and developers.

The general feeling amongst users and developers is that early user involvement is essential to produce a quality system that meets user needs. Developers widely acknowledge the importance of recruiting the appropriate user to maximise quality returns from UCD and this point is also reported in literature.

Summary of Approaches

UCD involves a change in development processes which is seen to improve system quality by the users and developers in this situation. The usability testing conducted on completed systems requires a minimal change in process from the developers and is seen by many as a form of acceptance testing. Systems that undergo testing without UCD are felt to incorporate less of the users requirements, are described as possessing a lower level of usability, and in some cases require re-engineering to achieve the required quality level for the organisation. UCD is seen to produce systems with a greater improvement in use quality that are more effective for use and are described as providing a superior support of users in the performance of their work. These systems are described as more flexible and easier to learn.

Developers from both the testing perspective and the UCD perspective appreciate the methods they use, and both perspectives feel that the benefits of usability practices is best demonstrated in the usability lab. In some cases, testing developers seemed reluctant to pursue UCD due to the perceived work involved and the change required in the development process. UCD developers felt that the effort was worth the reward. Both stated that usability practices were imperative for the production of a quality system.

CONCLUSIONS

The attributes described in the literature as indicative of system quality include flexibility, usefulness, reliability, effectiveness, efficiency, user satisfaction, ease of use, and ease of learning. These attributes are examined here in relation to process improvement and use improvement. It has been found that the late stage in system development that traditional usability testing occurs limits the quality benefits that can be obtained as major changes to the system are difficult. The minimal user participation that is involved in this practice also limits usability and system quality benefits. The UCD process involves users in development from the outset and users are able to specify their requirements for a quality system for their particular needs. There are disadvantages inherent in the process, however system quality is described by all parties as largely improved in systems developed using UCD.

This research has not sought to present quantitative data regarding usability testing and UCD. Some important questions remain unanswered, such as the quality impacts achievable through alternative usability testing practices including iterative testing. In order to determine the best methods for improving system quality using usability tools an evaluation of the full range of usability testing practices should be undertaken.

REFERENCES

- Borgholm, T., and Madsen, K. (1999). Cooperative Usability Practices, *Communications of the ACM*, 42(5), pp.91-97.
- Buur, J., and Bagger, K. (1999). Replacing Usability Testing with User Dialogue, *Communications of the ACM*, 42(5), pp.63-66.
- DeLone, W.H., and McLean, E.R. (1992). Information Systems Success: The Quest for the Dependent Variable, *Information Systems Research*, 3(1), pp. 60-95.
- Dolan, W., and Dumas, J. (1999). A Flexible Approach to Third-Party Usability, *Communications of the ACM*, 42(5), pp.83-85.
- Eriksson, I., and Torn, A. (1991). A model for IS quality, *Software Engineering Journal*, July, pp.152-158.
- Gardner, J. (1999). Strengthening the Focus on Users' Working Practices, *Communications of the ACM*, 42(5), pp.79-82.
- Garvin, D. (1984). What does "Product Quality" really mean?, *Sloan Management Review*, Fall, pp.25-39.
- Handy, C. (1985). *Understanding Organisations*, Penguin, London.
- ISO 8402:1994 Quality management and quality assurance.
- ISO 9241-11:1998 — Part 11: Guidance on usability.
- Muller, M., and Czersinski, M. (1999). Organising Usability Work to Fit the Full Product Range, *Communications of the ACM*, 42(5), pp.87-90.

- Myers, B., Kappelman, L., and Prybutok, V. (1997). A comprehensive model for assessing the quality and productivity of the information systems function, *Information Resources Management Journal*, Winter, pp.4-33.
- Ovaska, S. (1991). Usability as a Goal for the Design of Computer Systems, *Scandinavian Journal of Information Systems*, 3, pp.47-62.
- Potter, L. (1999). Usability and the User: An Exploration of the Relationship Between Usability and Information Systems Success, Honours Thesis, Griffith University.
- Reijonen, P., and Kesti, J. (1994). Information systems in use: from usability to exploitability, *Proceedings of the 17th IRIS*, pp.345-356.
- Salmela, H. (1997). From information systems quality to sustainable business quality, *Information and Software Technology*, 39(12), pp.819-825.
- von Hellens, L. (1997). Information systems quality versus software quality, *Information and Software Technology*, 39(12), pp.801-808.
- Vredenburg, K. (1999). Increasing Ease of Use, *Communications of the ACM*, 42(5), pp.67-71.
- Wilson, S., Bekker, M., Johnson, P., and Johnson, H. (1997). Helping and hindering user involvement - a tale of everyday design, *Proceedings from CHI97*, ACM, March, pp.1-14.

Chapter 16

Facilitating End User Database Development by Working with Users' Natural Representations of Data

Valerie J. Hobbs and Diarmuid J. Pigott
Murdoch University, Western Australia

One of the main advantages of user-developed applications is considered to be the greater familiarity the users themselves have with the problem domain, and hence the greater likelihood of their creating an application that meets their needs. However, it is equally frequently reported that many end users lack the skills to develop applications that are of a high quality. Database modelling and relational database design, in particular, are known to be problematic for novices. We present two case studies in which the first stage of the development process was completed entirely by the end user, making use of their own understanding of the dataset, the problem domain, and tools that were familiar to them. In each case, they had represented the data in the form of lists. An IT expert then facilitated the conversion of the dataset to a relational database, with the participation of the end users throughout the process. The end users were able to see the concepts of database design emerge naturally from a problem that was already familiar to them, and to understand their importance in a practical manner.

INTRODUCTION

There is a large body of literature on end user computing and on user-developed applications (UDAs). UDAs are applications developed by end users, rather than IT professionals, to undertake some particular task. The basic premise of end user development is that the users themselves are in the best position to understand the requirements of the application domain and therefore to create an application tailored to their particular needs. Thus, one of the often-quoted advantages of end user application development is that it eliminates the problematic step of user requirements elicitation in the systems development life cycle (Agboola, 1998). Indeed, Lally (1995) noted that a key issue in end user computing is determining how organizations can more effectively utilise the end user's superior knowledge of the application requirements in the process of application development.

However, there is also a large body of literature pointing out the potential hazards of UDAs that are developed when users without adequate background or training develop their own applications. Lack of familiarity with development methodologies, or with application software, may result in significant errors in the final product, despite their understanding of the requirements (Panko & Halverson, 1996). Agboola (1998) attempted to separate the effects of modelling knowledge and application domain knowledge in an experimental setting, and found support for the commonly held view that application domain knowledge was less important than modelling knowledge as a predictor of database implementation correctness.

Database management systems, today overwhelmingly personal relational systems such as Microsoft Access™, FoxPro™ or FileMaker Pro™, are one of the most common types of software used to create UDAs. However, normalised relational tables are not a natural way of representing data for most people. Further, relational model concepts such as keys, functional dependency and referential integrity, essential to create a correct and flexible database, are difficult to grasp. The spreadsheet format, where information is set out in tabular format, is generally thought to be more intuitive for novices, although is certainly not immune from errors (Panko & Halverson, 1996).

Hutchins, Hollan and Norman (1985) postulated the concept of "gulf" between a user's goals and the computer interface: if the gulf between the user's goals and the computer interface is large, the user will require more effort to accomplish their goals. This gulf is made up of *semantic* distance, which is the gulf between a user's conceptualisation of the "real world" object and an abstraction of it; and *articulatory* distance, which is the gulf between the meaning or abstraction of the object and its physical form or syntax. Batra (1993) further identified *mapping* (transforming the world into a representation), *rules* (which govern the mapping) and *consistency* (between the world and the representation) as distinc-

tions within semantic distance. Conceptual and logical database design, which involve translating the semantics of a problem into a data representation, are recognised to be difficult tasks for novice designers (Batra & Antony, 1994). This may be attributed to a large semantic distance between the user's view of the world and its representation as a data model or set of relational tables. Although current DBMS products such as Microsoft Access™ typically offer the end user a great deal of assistance in implementation of a design, relatively little help is offered in the earlier, modelling stage.

Work with novice and expert database designers has provided some insight into the types of errors that inexperienced data modellers typically make. Whereas novices can generally identify entities and attributes correctly, they frequently have problems with connectivity of relationships (e.g., Batra, Hoffer, & Bostrom, 1990), especially binary and ternary relationships. Batra (1993) reported that novices also had more problems representing binary relationships, particularly many-to-many, in relational tables than in Entity-Relationship diagrams.

Many authors (e.g., Kreie, Cronan, Pendley, & Renwick, 2000; Salchenberger, 1993) see training end users in analysis and design techniques as essential to improving UDA quality. Ahrens and Sankar (1993) recommended software tutors that teach specific database design skills. Batra and Antony (1994) suggested that an approach based on heuristics would be more efficient than one based solely on database principles. Batra (1993) developed a typology of database design errors, and suggested ways in which training strategies and knowledge-based tools could be developed to address particular types of errors. Agboola (1998) concluded that end users needed a large amount of hands-on database training to become competent, and suggested that collaborative development between end users and professionals could be a viable solution.

In this chapter, we describe two case studies of user-developed database applications that were facilitated by an IT professional. In each example, the users were experts in the application domain, and the data requirements were fully known. Both sets of users were computer literate, one with extensive spreadsheet experience, but neither had any database development experience and had called in one of the authors (DP) as a consultant when they realised database functionality would be required. The consultant made the decision at the outset to work directly with the users' own models of the data, and to convert them directly to database implementations with the active involvement of the users. One data set was in a series of spreadsheets, while the other was in the form of a table in a desktop publishing document. Throughout the process of conversion to the database the end users were instructed in the principles of database design, with relational model concepts emerging naturally from the need to keep the data consistent within its application domain. The aim of the consultant was to make use of the users' deep understanding of their particular dataset to facilitate their understanding of the

general database concepts. In this way it was hoped they would be left with the confidence and competence to be “self sufficient” in maintaining the database applications in future.

THE CASE STUDIES

Irish parish records

The first case study concerned historical research into the parish records for Killucan and Kinnegad in Co. Westmeath, Ireland. The research concerns “bringing back to life” a village that had lost most of its records, by assembling the sum of information available from various resources, including the baptismal registry and the occupancy of graveyards, Westmeath Grand Jury presentment books, and the vestry service records. A vast number of official records were destroyed during the Troubles in Ireland in 1922, making these isolated pockets of information highly important to genealogists.

The information had been collected in the field in notebooks by the historian, and had been transferred to embedded tables in Microsoft Publisher™ documents. Publisher was originally chosen as the research was to be included in a book on reconstructing the identities of defunct villages.

The challenge of reconstruction was to identify individuals and their lives from the many different records and sources. The database was needed to assist the historian in finding possible matches and discrepancies in the various forms of data, before the data was imported to a genealogical program using the GEDCOM transfer format, and to improve the accuracy of the tables in the original Publisher document. The historian further realised that database functionality would be needed to be able to make certain queries, such as listing all the known births and deaths in a particular parish, or matching up deaths of mothers with births of children or remarriage of fathers. Tracing the occupations of an individual as recorded throughout his life could also provide some measure of social mobility.

Environmental assessment

The second case study was a database of ecological data, collected by an environmental consulting company (Ecoscape) as part of a series of vegetation surveys of reserves for the Western Australian Department of Conservation and Land Management (CALM). Environmental assessment studies were to be carried out for three shires (Mount Marshall, Lake Bryde and Kent), including identification of vegetation units; the distribution of vascular flora; identification of rare or threatened plant taxa; identification of species and communities of significance; analysis of the relationships between soil, landform and vegetation characteristics; and assessment of vegetation vigour and health, particularly in relation to rising water tables and salinity.

Information was to be collected both at the level of the administrative unit (the Crown land reserve) and at the level of the unit of ecological sampling (the quadrat) within those reserves. At the quadrat level the information included soil chemistry, soil landform data, crown cover data, and vegetation species data. At the reserve level, the information included more general information, such as adjacent and previous land use, water features, presence of industry, road access and amenities, sites of cultural significance, and reserve intactness.

The data requirements were strictly prescribed in the tender documents and it was essential for the finished database to integrate with the CALM Herbarium species lists and the Department of Land Administration GIS (Geographic Information Systems) datasets as well as other pre-existing State databases. The terms of the tenders required the delivery of the information as Microsoft Access 97™ databases.

The consultant was first called in at the start of the project. Since the users were very familiar with spreadsheets, but had no database experience at all, the consultant made the decision to do all the initial data input using spreadsheets. The worksheet format also had the advantage of more directly reflecting the data collection process (on field data sheets), and permitted a degree of flexibility in structuring the data. This was desirable since Ecoscape's experience had shown that various ambiguities and new requirements in field data were likely to necessitate the alteration of the worksheets on an ad hoc basis as the project progressed. It also enabled Ecoscape to settle on their own precision values for numbers (e.g., integer, float, etc.) and modify them on the fly, rather than having to define fixed data types in advance.

The Ecoscape project team designed and constructed the spreadsheets themselves in Microsoft Excel 97™, and filled them in using the data from the collection sheets. At this stage the consultant was called back to facilitate the conversion of these spreadsheets into a database format.

DEVELOPING THE DATABASES

The Process

The conversion of the lists to a relational database involved several stages. We summarise these stages next, before describing the details of the process for each of the case studies.

- **Step 1: Discussion.** The process commenced with an informal discussion between the consultant and the end users. The users described their data and the sorts of processing they needed to undertake. The consultant gave the users a brief outline of relational databases to explain how their lists would eventually become a set of tables that could be used for querying and reporting. This explanation included constructing a rough sketch of the proposed tables and

the relationships between them, to gain feedback from the users. The discussion stage was vital to ensure the users could clarify any issues with the data with the consultant and correct any misapprehensions he had, and for the users to be prepared for the remainder of the process.

- **Step 2: “Cleaning” the data.** This involved checking that all values were correct, and checking the interpretation of empty values in the lists (i.e., whether they represented “no value,” “value not known,” or a match of the value above). “No value” cells were labelled “n.v.” and “not known” were labelled “n.k.” This step was done in spreadsheet format for ease of data manipulation.
- **Step 3: Creating the equivalent of a First Normal Form relation** (Date, 1991) in the spreadsheet. This was done by using Fill Down where there was repeating data to ensure every cell had a value.
- **Step 4: Creating and populating the set of normalised relations** (Date, 1991) in the database, directly from the spreadsheet. This was done table by table, starting with “parent” tables, and involved:
 - Copying one table’s worth of data at a time to the database from the spreadsheet, to create a new table
 - Creating the primary key of the table
 - Creating (where necessary) the appropriate foreign key values from the data in previously created tables
 - Setting referential integrity between the new table and previously created tables
 - Altering the field length or data types where necessary
- **Step 5: Reconstitution.** Once all the tables were created and populated, queries were made to check that the normalisation had been lossless (Date, 1991) and to demonstrate to the user that all of the original data was still present in the normalised tables.
- **Step 6: Reviewing and formalising the database design process with the user.** The consultant used the Relationships window in Access to demonstrate to the user how the final tables linked together, and to discuss the concepts of keys and referential integrity more formally. The consultant then returned to the concept of data modelling using Entity-Relationship diagrams (Chen, 1976), introduced informally in Step 1, and demonstrated how a formal ERD would be drawn for the system just constructed. This led to discussion of how data modelling is used in the early stages of database design, and how the resulting relational design facilitates querying and reporting.

IRISH PARISH RECORDS DATABASE

The original data was in the form of tables embedded in a set of Microsoft Publisher documents, each document listing the records for a source of informa-

Figure 1: Microsoft Publisher document with table of baptism records.

KINNEGAD FAMILIES WITH BAPTISMS AT KILLUCAN 1696-1778

Family Name	Father	Occupation	Wife	Children	Baptism Dates
ADAIR					
ASHE	Richard				
ASHE	Robert		Elizabeth GARBETT		
AUSTIN	William	Burchar	Alise	William	27-Sep-1750
AUSTIN	William	Alexander	Elizabeth	Margary	12-Aug-1745
BELL	Turner			George	19-Nov-1759
BELTON	Thomas			Richard	15-May-1768
BEST	Richard		Catherine	Richard	18-Jun-1768
BEST	Richard		Catherine	Rice	19-Sep-1775
BEST	Richard		Sharon RICHARDS		
BRAZER	George	Bridgewater	Elizabeth	Edwin	04-Apr-1779
BRAZER	George	Bridgewater	Elizabeth	John	20-Feb-1789
BRAZER	George	Bridgewater	Elizabeth	William	01-Feb-1790
BRAZER	George	Bridgewater	Elizabeth	Stanza	12-May-1743
BRAZER	George	Calix Maker	Elizabeth	Jane	18-Jun-1744
BRAZER	George	Harvey Maker	Elizabeth	Elsner	10-Sep-1745

tion for a particular parish. The tables read left to right, with blank records indicating a repeat of the information listed above. Figure 1 shows part of the table for baptismal records from Killucan parish.

The process of converting to a database was then carried out following the steps described above.

• Step 1: Discussion

Several issues relating to the application domain were resolved during the initial discussion between the historian and the consultant. It was decided not to include the information for the vestry records and gravestones, since there was so little available, and to use only the baptism records. The consultant and historian discussed the possibility of recording multiple marriage in the database, but made the decision not to as the sources only recorded multiple marriages for men but not women. The absence of any remarriages in the small data set meant that a many-to-many relationship between the wives and husbands tables was avoided.

A preliminary diagram showing the proposed entities in the database was drawn and validated with the end-user. These entities were Fathers, Mothers, and Children.

• Step 2: Data cleaning

The original tables were copied from Publisher into Excel to facilitate data cleaning. All the baptism records from the two parishes were copied into a single worksheet, with an extra column "Parish" added to record the parish name implicit in the original document titles. As some wives had surnames as well as first names recorded a column "Wife other surname" was added to the worksheet,

Figure 2: 'First normal form' data for Killucan baptism records.

	A	B	C	D	E	F	G	H
	Parish	Family Name	Father	Occupation	Wife	Wife other name	Children	Baptism Dates
2	Killucan	ASHE	Robert	n.k.	n.k.		n.k.	n.k.
3	Killucan	AUSTIN	William	n.k.	n.k.		n.k.	n.k.
4	Killucan	BRAIZER	George	n.k.	Elizabeth	GARBETT	n.k.	n.k.
5	Killucan	BRAIZER	George	Butcher	Alice		William	27-Sep- 1750
6	Killucan	BRAIZER	George	Alesaller	Elizabeth		Margary	12-Aug- 1745
7	Killucan	BRAIZER	George	n.k.	n.k.		George	19-Nov- 1759
8	Killucan	BRAIZER	George	n.k.	n.k.		Richard	15-May- 1703
9	Killucan	BRAIZER	George	n.k.	Catherine		Richard	18-Jun- 1705
10	Killucan	BURGESS	n.k.	n.k.	Catherine		Alice	19-May- 1705
11	Killucan	CALDWELL	William	n.k.	Suzanne	RICHARDS	n.k.	n.k.
12	Killucan	CALDWELL	William	Bridlecutter	Elizabeth		Esther	04-Apr- 1729
13	Killucan	CALDWELL	William	Bridlecutter	Elizabeth		John	20-Feb- 1739
14	Killucan	CODD	George	Bridlecutter	Elizabeth		William	01-Feb- 1740
15	Killucan	CODD	George	Bridlecutter	Elizabeth		Eleanor	12-May- 1743
16	Killucan	CODD	George	Collar Maker	Elizabeth		Jane	18-Jun- 1744
17	Killucan	CODD	George	Harness Maker dec'd	Elizabeth		Eleanor	10-Sep- 1745
18	Killucan	CODD	George	Joiner	Eleanor		Samuel	30-Aug- 1709
19	Killucan	CODD	George	Joiner	Eleanor		Matthew	12-Feb- 1710
20	Killucan	CODD	George	Joiner	Eleanor		Edward	26-Oct- 1713
21	Killucan	COYNE	Cornelius	Weaver	Alice		John	10-Jul- 1715

and the surnames moved to that column from the "Wife" column. Unknown or missing data was given "n.k." values.

- **Step 3: First Normal Form**

All empty cells that indicated repeating data were filled in. Figure 2 shows the data set at the completion of this step.

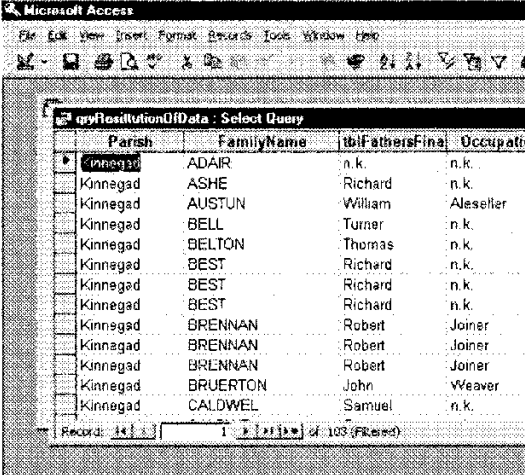
- **Step 4: Normalised tables**

The first table created was "Fathers." A new table was created in Access Datasheet view, and the contents of the worksheet columns "Parish," "Family name," "Father" and "Occupation" were pasted into this table using Paste Append. The fields were given the same name as the worksheet column names. From this temporary table the new table called "Fathers" was created from a make-table query retrieving all fields, using SELECT DISTINCT to ensure no repeating data. An AutoNumber field was added as primary key.

A table called "Mothers" was created in a similar manner, based on the contents of the columns "Parish," "Family name," "Father," "Occupation," "Wife" and "Wife other name" in the worksheet, with an Autonumber primary key. The names of the columns in the worksheets were used for the field names. The field data types were all left as text.

A field "HusbandFK" was added to the Mothers table to act as a foreign key to the Fathers table. The values for HusbandFK were added through an update query matching the Mothers and Fathers tables on the fields "Parish," "Family name," "Father" and "Occupation." The relationship between the two tables was created in the Relationships window, and referential integrity enforced. There were no errors. The fields "Parish," "Father" and "Occupation" were then deleted from the Mothers table, since they were only required for generating the foreign key

Figure 3: Query rejoining all records.



The screenshot shows the Microsoft Access interface with a query window titled 'qryResitutionOfData: Select Query'. The query results are displayed in a table with the following columns: Parish, FamilyName, tblFathersFina, and Occupatio. The data is as follows:

Parish	FamilyName	tblFathersFina	Occupatio
Kinnegad	ADAIR	n.k.	n.k.
Kinnegad	ASHE	Richard	n.k.
Kinnegad	AUSTUN	William	Alaseller
Kinnegad	BELL	Turner	n.k.
Kinnegad	BELTON	Thomas	n.k.
Kinnegad	BEST	Richard	n.k.
Kinnegad	BEST	Richard	n.k.
Kinnegad	BEST	Richard	n.k.
Kinnegad	BRENNAN	Robert	Joiner
Kinnegad	BRENNAN	Robert	Joiner
Kinnegad	BRENNAN	Robert	Joiner
Kinnegad	BRUERTON	John	Weaver
Kinnegad	CALDWEL	Samuel	n.k.

At the bottom of the window, it indicates 'Records: 14 of 14' and '1 of 103 (Filtered)'.

values. The table was deliberately kept in 2NF, as it was decided to retain the field “Family name” for ease of display.

Finally, a table called “Children” was created. This table was based on the spreadsheet columns “Parish,” “Family name,” “Father,” “Occupation,” “Wife,” “Children” and “Baptism dates.” As the baptism of a child is a unique event there was no need for the SELECT DISTINCT query here. Two new fields “FatherFK” and “MotherFK” were added to act as foreign keys to the Fathers table and Mothers table respectively. The fields FatherFK and “MotherFK” were populated using an update query as described above, and the redundant fields removed from the Children table. Again, the field “Family Name” was kept for the sake of convenience. Referential integrity was enforced between Children and Mothers, and Children and Fathers, with no errors. An AutoNumber field was added as a primary key, as there were 13 first and family name pairs repeated at least once in the dataset.

Figure 4: The Relationships window in Access showing final table design for the Parish records database.

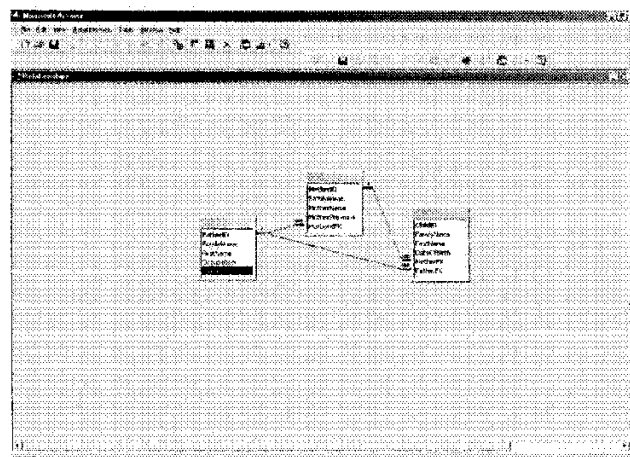


Figure 5: Extract from a blank field collection sheet for Mount Marshall. This part of the sheet records quadrat soil data.

Date	Reserve #	Quadrat #	MIM	Collector
Soil Sample #				
Soil Group				
Code				
SOIL PROFILES TO MAX. DEPTH OF 1m				
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p>HORIZON A</p> <p>Type: <input type="text"/></p> <p>Depth of horizon: <input type="text"/> cm</p> <p>Upper depth (m): <input type="text"/></p> <p>Lower depth (m): <input type="text"/></p> <p>Existing vertical exposure: <input type="text"/></p> <p>Correlatively undisturbed soil core: <input type="text"/></p> <p>Auger boring: <input type="text"/></p> <p>Boundary description: <input type="text"/></p> <p>Structure: <input type="text"/></p> <p>Field texture grade: <input type="text"/></p> <p>Soil glassiness: <input type="text"/></p> <p>Soil water regime: <input type="text"/></p> <p>Notes: <input type="text"/></p> </div> <div style="width: 48%;"> <p>HORIZON B</p> <p>Type: <input type="text"/></p> <p>Depth of horizon: <input type="text"/> cm</p> <p>Upper depth (m): <input type="text"/></p> <p>Lower depth (m): <input type="text"/></p> <p>Existing vertical exposure: <input type="text"/></p> <p>Correlatively undisturbed soil core: <input type="text"/></p> <p>Auger boring: <input type="text"/></p> <p>Boundary description: <input type="text"/></p> <p>Structure: <input type="text"/></p> <p>Field texture grade: <input type="text"/></p> <p>Soil glassiness: <input type="text"/></p> <p>Soil water regime: <input type="text"/></p> <p>Notes: <input type="text"/></p> </div> </div>				
<div style="display: flex; justify-content: space-between;"> <div style="width: 48%;"> <p>COARSE FRAGMENTS</p> <p>Abundance: <input type="text"/></p> <p>Size: <input type="text"/></p> <p>Significance of: <input type="text"/></p> <p>Notes: <input type="text"/></p> </div> <div style="width: 48%;"> <p>COARSE FRAGMENTS</p> <p>Abundance: <input type="text"/></p> <p>Size: <input type="text"/></p> <p>Significance of: <input type="text"/></p> <p>Notes: <input type="text"/></p> </div> </div>				

• Step 5: Reconstitution

A query was then made to rejoin all records to show that there had been no loss of data from the original tabular data set (Figure 3). Filters were used to show the original two lists (Kinnegad and Killucan) as logically separate.

Several other queries were demonstrated to the user, based on her requirements. For example, the "Find Duplicates" query wizard was used to show repeated children's name pairs, and to point out possible matches between people and occupations.

• Step 6: Review

The Access Relationships window was used to demonstrate how the final state of the database matched the rough diagram discussed with the historian at the start of the process (Figure 4).

Environmental assessment database

Prior to the consultancy, Ecoscape had designed the field data collection sheets in accordance with the tender documents. These data collection sheets were required to be kept as a permanent record. The data collection sheets were designed in sections, each section grouping together the different information required to be collected under the terms of the contract (Figure 5).

As the completed field sheets came back from the survey teams, together with the samples and site photographs, the data they contained was transferred to the spreadsheets designed by the Ecoscape project team. There were three separate Microsoft Excel 97 workbooks, one for each project (shire). Each workbook contained a number of worksheets for the different types of data collected. These worksheets corresponded to the sections on the field data collection sheets.

Figure 6: The 18 worksheets of data collected for each shire.

- Reserve details and descriptions (Figure 7)
- Reserve intactness
- Reserve water resources
- Reserve environmental weeds (Figure 8)
- Reserve rare flora
- Reserve commercial land use
- Reserve adjacent land use
- Reserve fauna
- Reserve amenities/tourism facilities
- Reserve sites of cultural significance
- Reserve PIN data
- Quadrat details and descriptions (Figure 9)
- Quadrat vegetation descriptions (Figure 10)
- Quadrat crown cover
- Quadrat crown cover gap and width measurements
- Quadrat soil and landform details
- Quadrat soil profile
- Quadrat soil chemistry

Figure 7: Reserves details worksheet for Mt Marshall.

Reserve No	Polygon IDNo	Reserve Name	Lot Number	Location No	Shire	Land District	District Name	District No	Reserve Area
12689	741586	Waircubbing Nature Reserve		14805	Mt Marshall	Avon	Merredin	33	73
13509	741749			14281	Mt Marshall	Avon	Merredin	33	135
25323	738218			3241	Mt Marshall	Ninghan	Merredin	33	178
28486	741333			4174	Mt Marshall	Ninghan	Merredin	33	8
14642	741385			1353	Mt Marshall	Ninghan	Merredin	33	40
15437	741434			1596	Mt Marshall	Ninghan	Merredin	33	20
16423	738583			20430	Mt Marshall	Avon	Merredin	33	5
39186	738176			3000	Mt Marshall	Ninghan	Merredin	33	103
39186	989524				Mt Marshall	Ninghan	Merredin	33	

Figure 8: Distribution of weeds in reserves for Mount Marshall.

Reserve No	Reserve Name	Weed Sp. Code	Weed Species	Weed Infestation	Comments
12689	Waircubbing Nature Reserve	ERHCAL	Eriarta calycina	4	Understorey largely comprised of weeds
12689	Waircubbing Nature Reserve	SOLHOP	Solanum hoplopetalum	4	Understorey largely comprised of weeds
VCL02	Vacant Crown Land 2	ECHPLA	Echium plantagineum	2	Weed cover patchy
VCL02	Vacant Crown Land 2	PLASP1	Plantago sp1	2	Weed cover patchy
VCL02	Vacant Crown Land 2	BRATOU	Brassica tournefortii	2	Weed cover patchy

Figure 9: Quadrat details worksheet for Mount Marshall.

Date	Collector	Reserve No	Polygon IDNo	Quadrat No	Datum	Zone	Easting	Northing	Accuracy	Duration
13.05.00	WEGJ	25323	738218	MM0001	AGD84	50	578786	6653493	5.3	80
13.05.00	WEGJ	25323	738218	MM0002	AGD84	50	578127	6653870	4.7	45
13.05.00	WEGJ	25323	738218	MM0003	AGD84	50	578309	6653824	3.8	60
15.05.00	WEGJ	25323	738218	MM0004	AGD84	50	577999	6653261	3.6	45
11.05.00	WEGJ	39186	989524	MM0005	AGD84	50	562780	6639424	3.7	45
11.05.00	WEGJ	39186	738176	MM0006	AGD84	50	563905	6639666	4.2	55
17.05.00	WEGJ	VCL01	741143	MM0007	AGD84	50	595206	6641273	4.0	10
17.05.00	WEGJ	VCL01	741143	MM0008	AGD84	50	594754	6641363	4.0	40

Figure 10: Plant species for quadrats in Mount Marshall.

Quadrat No	Species Name	Species Code	Voucher Number	Stratum	Growth Form	Average Height
MM0001	<i>Eucalyptus celestroides</i>	EUCCEL	MMV0074	1	M	12.0
MM0001	<i>Eucalyptus loxophleba</i> subsp. <i>supraelevis</i>	EUCLOXSUP		1	M	10.0
MM0001	<i>Acacia tetragynophylla</i>	ACATET	MMV0075	2	S	2.1
MM0001	<i>Acacia affinis</i>	ACAERI		3	S	0.6
MM0001	<i>Grevillea acutata</i>	GREACU	MMV0076	3	S	0.5
MM0001	<i>Olearia muelleri</i>	OLEMEU		3	S	0.5
MM0001	<i>Phyllanthus obovatus</i>	PTIORO	MMV0077	3	S	0.3
MM0001	<i>Rhus glabra</i>	RHADRU		3	S	0.4
MM0001	<i>Sclerolaena dicrantha</i>	SCLDIC		3	C	0.1
MM0001	<i>Austrodanthonia</i> sp1	AUSSP1		4	G	0.2
MM0002	<i>Acacia resinomarginata</i>	ACARES	MMV0078	1	T	5.5
MM0002	<i>Acacia stereophylla</i> subsp. <i>stereophylla</i>	ACASTE	MMV0080	1	T	3.5
MM0002	<i>Eucalyptus leptopoda</i> subsp. <i>subulata</i>	EUCLEPSUB		1	M	4.5
MM0002	<i>Eucalyptus ordii</i>	EUCOLD	MMV0081	1	M	4.0

The worksheets are listed in Figure 6 and examples of data shown in Figures 7-10. Several of the worksheets recording reserve data had more than one record for each reserve (e.g., there were many fauna records for each reserve). Similarly, several quadrats had multiple records (e.g., there was more than one vegetation species per quadrat (Figure 10)).

The final dataset contained 2355 data points about 152 quadrats in 56 reserves. There were 946 data points collected about the reserves themselves.

The consultant then followed the same process to develop the Access database from the spreadsheets, with the participation of the Ecoscape project team.

- **Step 1: Discussion**

The initial discussion identified a likely set of tables and the relationships between these tables were sketched. Since the information to be collected for each shire was largely the same and the same worksheets were found within each workbook, the data contained within the three workbooks could be combined in a uniform fashion.

It was clear that the main entities in the planned database were the “Project” (drawn from each workbook), “Reserve” (drawn from the Reserve Details worksheet in each work book) and “Quadrat” (drawn from the Quadrat Details worksheet in each workbook). The basic structure of the database was sketched showing each Project having many Reserves, and each Reserve having many Quadrats, and confirmed with the project team.

The names of the worksheets gave a clear indication of the levels at which the information was collected, and the sheets themselves were the obvious candidates for tables. Some were in a one-to one but optional relationship with the reserves (e.g., commercial land use) or quadrats (e.g., photos), others were in a many to one (e.g., fauna for reserves, flora for quadrats). The difference between these types of relationships was discussed, together with the necessity for a parent record (i.e., a reserve or a quadrat) for each value held. (This led to some complications for interpolated data for some reserves at quadrat level in the data cleaning stage.)

The proposed tables containing the data from these worksheets were added to the diagram and confirmed with the team.

- **Step 2: Data cleaning**

To record vegetation type data, a quadrat had to be created for each vegetation type for each reserve. Where this had not been done in the field (i.e., where the vegetation type had not been noticeably different and so had been missed) the information was added by interpolation using a GIS system. Vegetation type had already been added manually into the spreadsheet. This had necessitated the creation of “nominal quadrats” for vegetation data – uninvestigated areas of the Reserves that would be targets for inspection on the next field trip. Nominal quadrats records were therefore included in the “Quadrat Descriptions” worksheets.

The nature of blank cells was resolved (in consultation with Ecoscape) into those representing repeating data from the row above and those representing “no value.”

- **Step 3: First normal form**

A “first normal form” worksheet was created from each of the original worksheets, using Fill Down function to ensure every cell had a value. An extra column was added to the “Reserve Details” worksheet in each of the workbooks, to hold the values for the Project IDs.

- **Step 4: Normalised tables**

The first table to be created was “Projects,” to hold the information about the three shires. This data consisted of a code according to the CALM identification system, a reference title, and an automatically generated primary key. This table is parent to all the other tables.

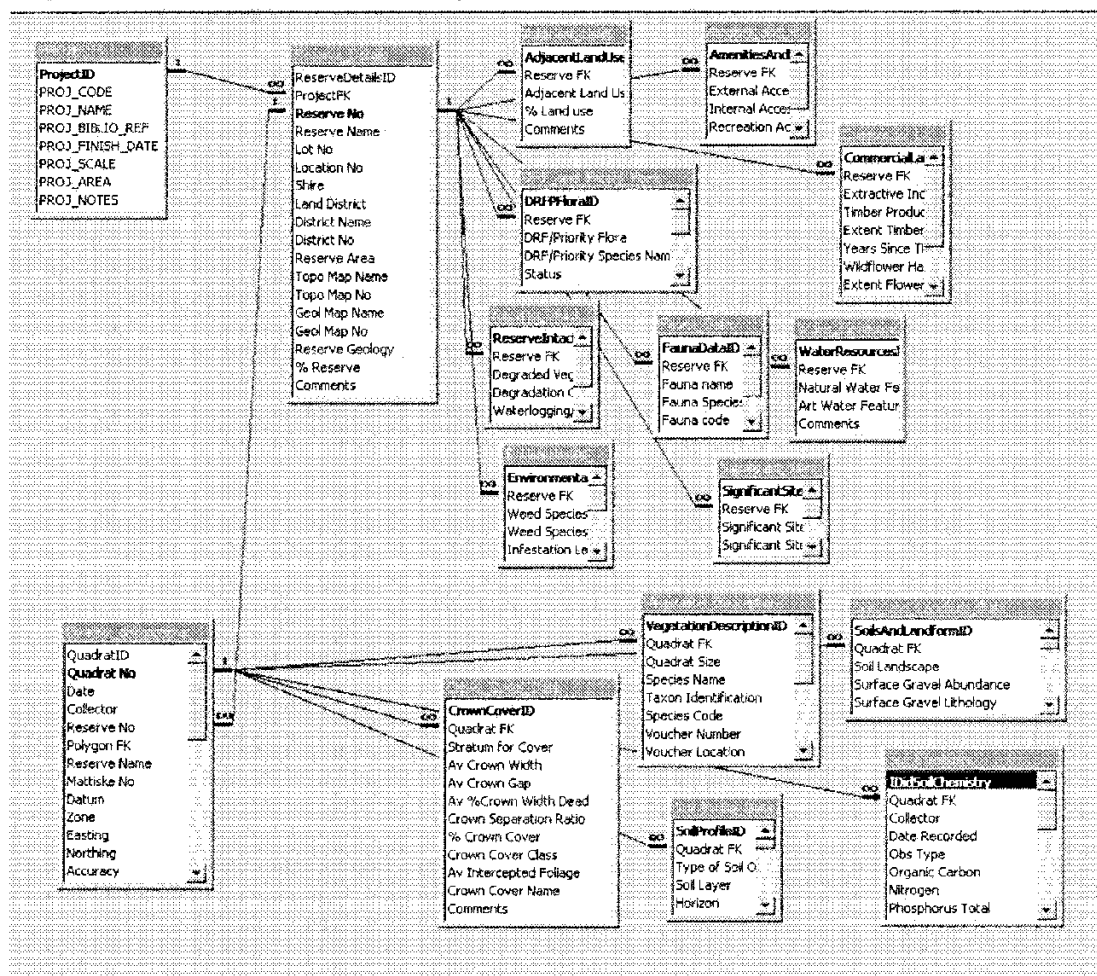
The data for Reserve Details and Descriptions was used to create a new table “Reserve Details.” The table was based directly on the spreadsheet data “Reserve Descriptions” as laid out, with the addition of a primary key for the table, and a foreign key to the project the reserve was part of. The fields were given the appropriate names according to the column headings in the workbook, and the table saved. Data for each of the three shires was added to the “Reserve Details” table in the same manner. Referential integrity was set with the “Projects” table.

All the other worksheets for information pertaining to reserves were then converted into tables one at a time (i.e., the worksheet for “Water Resources” became a table called “WaterResources,” and so forth), following the same process. As each table was created and populated, referential integrity was enforced using the relationship manager. In the event that it was not possible to enforce referential integrity, the Find Unmatched Query Wizard was used to isolate the offending record. There were very few exceptions (about 5 for the entire project). These exceptions gave the consultant the opportunity to explain the importance of referential integrity in a very practical way.

Figure 11: Union query for entire data set at reserve level.

Reserve FK	txtLabel	txtCol1	txtCol2	txtCol3	txtCol4
10685	DRF				
10685	Intactness				
10685	Significant site				
10685	Water Resources		1		
10685	Water Resources		2		
10685	Water Resources	1			
10685	Sandalwood	1			
10685	Amenities	1	2		
10685	Fauna	Australian Magpie	Gymnorhina tibicen	GYMTIB	V
10685	Fauna	Australian Ringneck	Barnardius zonarius	BARZON	V
10685	Fauna	Bobtail	Tiliqua rugosa	TILRUG	B
10685	ALU	cropping/grazing	74%		
10685	Weeds	EHRCAL	Ehrharta calycina	2	

Figure 12: Final table design for environmental assessment database.



The “Quadrat” table was created next, using the same process as described for Reserves. Referential integrity for the quadrat-reserve relationship was ensured as the Reserve ID was already in place for each row in the Quadrat worksheets.

Finally, tables corresponding to the different information collected in Quadrats were created in the same manner as the various tables containing reserve information, and referential integrity set.

In some instances the data after normalising was multivalued: e.g., “traces of fauna” in a quadrat could record several traces - droppings, marks, carcasses etc. Based on the extent of its occurrence and likely significance in the system, the consultant and Ecoscape made the decision to represent the information simply as a list, rather than normalising further.

• Step 5: Reconstitution

The conversion process was finalised by creating queries to reconstitute each of the original 1NF worksheets. Based on these queries, a single union query was created to present all of the data at quadrat level in a single data set, and another

Figure 13a: The conventional database development process.

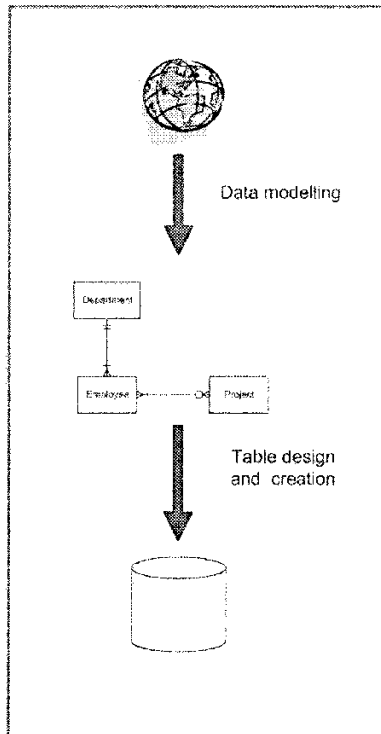
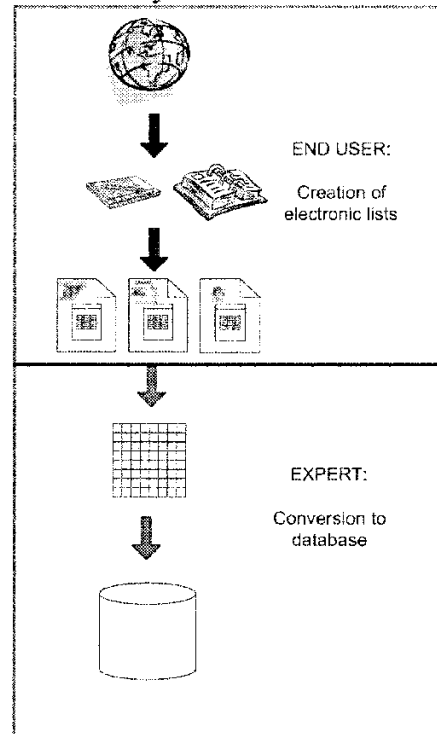


Figure 13b: The database development approach followed in this study.



to provide all of the data at reserve level. Figure 11 shows part of the union query integrating all of the data at reserve level (for example, the last line visible in the table is derived from the worksheet of reserve weeds shown in Figure 8). This format was required by CALM in the terms of the tender, to interface with the GIS inquiry agent.

- **Step 6: Review**

The Access Relationships window was then used to show how the original proposed database diagram had been successfully implemented (Figure 12).

DISCUSSION

If the end users had followed a conventional process of database development (e.g., Connolly & Begg, 1998), they would most likely have analysed and modelled the data requirements (probably in an ERD), created the tables in the DBMS, and entered the data from the field collection sheets or notebooks (Figure 13a). As discussed earlier, this process is likely to involve a large semantic and syntactic “gulf” (Batra, 1993) between the user’s model of reality and the database implementation, with the possibility of errors and a poor quality product.

Instead, a two-stage process was followed. First, the end users developed their own systems for recording the required data in list format, using software with which they were familiar. In the second stage, the lists were converted into a

database by the consultant (Figure 13b).

The end users and the consultant were each able to work within their own area of expertise, reducing the semantic or syntactic distance they had to deal with. The end users, with their expert knowledge of the data, were able to work directly with the data and construct data sets that represented their view of the world in a very naturalistic manner. When the consultant entered the process, the data set was completely known – thus there were no problems of requirements elicitation apart from the clarification of a few minor points.

The facilitated process of converting from list format to database allowed relational database principles to be demonstrated to the users using data about which they were experts. We noted the natural emergence of abstractions from the data, which illustrated such concepts as entity, attribute, relationship, normalisation and referential integrity. For example, we found that the environmental assessment data had effectively been normalised by Ecoscape in the process of identifying the data collection points and recording in the worksheets, and the consultant had only to formalise the principle.

The consultant was able to follow a straightforward, structured process of converting the lists into a relational database. A factor in this was undoubtedly the help provided by the integration of the software packages used. Copying and pasting records from Excel to Access permitted an almost seamless transition between the spreadsheet and the database view of data. The ability of Access to provide autonumber keys, and the graphical representation of referential integrity, also made the definition of primary keys and foreign keys very straightforward. The Relationships window was valuable in helping the end users to see the ‘total’ picture and appreciate how queries and reports could be constructed.

The list format appears to be a more natural way of representing data for people unfamiliar with dealing with abstractions of data and logical relationships between them. The approach described here should be applicable to any other problem domain in which the end user is able to represent their data in the form of a list of facts with repeating values. (It should be noted that we are discussing only application domains suitable for representing as databases: “lists” requiring true spreadsheet functionality such as calculations would not be suitable.) The use of the list format as an intermediate modelling stage has inherent advantages in some situations: for example it was particularly suited to the fluid nature of the scientific data collected by the Ecoscape team.

Some minor disadvantages to the process were noted. Ecoscape had made use of cell comments, and this information was lost when the spreadsheets were converted. Any information about sequence implicit in the spreadsheet format would also be lost, as would intermediate data where only the final version of a derived value was copied (although neither applied in the examples here). To

retain these sorts of information would be possible, but would involve a less straightforward process of conversion.

We have not yet been able to follow up the progress of either the Ecoscape team or the historian, but it would be interesting to discover whether they are able to apply their new database skills to novel situations, such as continuing maintenance of this project, design and implementation of other projects, or (in the case of Ecoscape) instructing new members of their team. The participation of the end user and the explanation of the conversion process may impart sufficient understanding and skills for them to attempt similar projects in the future. However, it may be that the facilitation offered by the expert is an essential part of the process.

We believe that this approach of collaboration between end user and professional, making the best use of their separate areas of expertise, has great potential both to ensure high quality of database development and to provide appropriate training in database design for end users. The end users in the case studies reported were left with a sense of ownership of the process of design, and confidence in their ability to work with the finished product.

ACKNOWLEDGMENTS

We are grateful to David Kaesehagan, Sandy Griffin and Alan Hunter of Ecoscape for permission to use the Reserve case study, and to Sandra Pigott for use of the Parish records project.

REFERENCES

- Agboola, I. O. (1998). *An experimental study of the effects of application domain knowledge and data modeling knowledge on the quality of database implementations by non-experts*. Unpublished PhD, University of Maryland.
- Ahrens, J. D., & Sankar, C. S. (1993). Tailoring database training for end users. *MIS Quarterly*, 17(4), 419-439.
- Batra, D. (1993). A framework for studying human error behavior in conceptual database modeling. *Information & Management*, 25, 121-131.
- Batra, D., & Antony, S. R. (1994). Novice errors in conceptual database design. *European Journal of Information Systems*, 3(3), 57-69.
- Batra, D., Hoffer, J. A., & Bostrom, R. P. (1990). Comparing representations with relational and EER models. *Communications of the ACM*, 33(2), 126-139.
- Chen, P. P.-S. (1976). The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, 1(1), 9-36.
- Connolly, T. M., & Begg, C. E. (1998). *Database Systems: A Practical Approach to Design, Implementation and Management*. (2nd ed.): Addison

Wesley Longman.

Date, C. J. (1991). *An Introduction to Database Systems*. (5th ed.). (Vol. 1): Addison-Wesley.

Hutchins, E. L., Hollan, J. D., & Norman, D. A. (1985). Direct manipulation interfaces. *Human-computer interaction*, 1, 311-338.

Kreie, J., Cronan, T. P., Pendley, J., & Renwick, J. S. (2000). Applications development by end-users: Can quality be improved? *Decision Support Systems*, 29(2), 143-152.

Lally, L. (1995). Supporting appropriate user-developed applications: Guidelines for managers. *Journal of End User Computing*, 7(3), 3-10.

Panko, R. R., & Halverson, R. P. (1996). Spreadsheets on trial: A survey of research on spreadsheet risks. *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, 2, 326-335.

Salchenberger, L. (1993). Structured development techniques for user-developed systems. *Information & Management*, 24, 41-50.

Chapter 17

User Developed Applications: Can End Users Assess Quality?

Tanya J. McGill
School of IT
Murdoch University, Australia

Organizations rely heavily on applications developed by end users yet lack of experience and training may compromise the ability of end users to make objective judgments about the quality of their applications. This study investigated the ability of end users to assess the quality of applications they develop. The results confirm that there are differences between the system quality assessments of end user developers and independent expert assessors. In particular, the results of this study suggest that end users with little experience may erroneously consider the applications they develop to be of high quality. Some implications of these results are discussed.

INTRODUCTION

User developed applications (UDAs) form a significant proportion of organizational information systems (IS) (McLean, Kappelman, & Thompson, 1993) and the ability to use end user development tools is often a position requirement instead of an individual option (Brancheau & Brown, 1993). The benefits that have been claimed for user development of applications include better access to information and improved quality of information, leading to improved employee productivity and performance. However the realization of these benefits may be put at risk because of problems with information produced by UDAs that may be incorrect in design, inadequately tested, and poorly maintained.

Despite these risks organizations generally undertake little formal evaluation of the quality of applications developed by end users (Panko & Halverson, 1996).

In the majority of organizations the only measures of whether an application is suitable for use are user developers' subjective assessments of their applications. Yet purely subjective, personal evaluations of UDA quality could be at wide variance with actual quality. Lack of experience and training may compromise the ability of end users to make objective judgments about the quality of their applications, but it appears that many end users do lack experience and training in both use of system development tools and in systems development procedures (Cragg & King, 1993).

There has been little empirical research on user development of applications (Shayo, Guthrie, & Igbaria, 1999), and most of what has been undertaken has used user satisfaction as the measure of success because of the lack of objective measures available (Etezadi-Amoli & Farhoomand, 1996). The fact that vital organizational decision making relies on the individual end user's assessment of application effectiveness suggests that more insight is needed into the ability of end users to assess the success of their own applications, and that as well as user satisfaction additional criteria of success should be considered.

Research on the relationship between experience or training and the success of UDAs has been inconclusive. In a meta-analysis of 15 end user satisfaction studies, Mahmood and Burn (1998) found that in the majority of studies greater levels of user developer experience were associated with higher levels of satisfaction. However individual studies vary: Al-Shawaf (1993) did not find any relationship between development experience and user satisfaction, while Amoroso (1986) found that the lower the level of programming skills and report building skills reported the higher was the satisfaction. Janvrin and Morrison (1996) found that their more experienced subjects were less confident that their applications were error free. Crawford (1986) found that higher levels of training were generally associated with lower levels of user satisfaction, while Raymond and Bergeron (1992) found microcomputer training to have a significant effect on satisfaction with decision making, and Nelson and Cheney (1987) concluded that there is generally a positive relationship between computer-related training that a user receives and his or her ability to use the computer resource. Yaverbaum and Nosek (1992) speculated that computer training increases one's expectations of information systems, and hence may actually cause negative perceptions. This may be the case for both training and experience in the UDA domain and may go some way to explaining the lack of conclusive results in the literature.

There have been many calls for the development of more direct and objective measures of UDA effectiveness (Al-Shawaf, 1993; Edberg & Bowman, 1996; Igbaria, 1990; Rivard, Poirier, Raymond, & Bergeron, 1997). There have been some attempts to move away from the use of user satisfaction as the major indicator of UDA success and to adopt a software engineering approach with a focus

on application quality rather than user satisfaction. Edberg and Bowman (1996) compared the quality of UDAs with applications developed by IS professionals, and found UDAs to be of significantly lower quality. Rivard and her colleagues (Rivard et al., 1997) noted that although the conceptual definitions of quality from the software engineering literature are appropriate for UDAs, the operationalizations in terms of software metrics are not. They therefore attempted to capture both the user perspective and the more technical aspects of UDA quality through a validated assessment instrument to be completed by end user developers (Rivard et al., 1997). However, none of these studies have compared user and expert assessments of UDA quality, nor looked at the roles of experience and training in end users ability to assess the quality of applications. This paper describes a study which uses direct examination of applications to compare users' and experts' assessments of user developed applications.

RESEARCH QUESTIONS

As discussed above, reliance on end user perceptions of UDA quality may be problematic because users may not only lack the skills to develop quality applications but may also lack the knowledge to make realistic determinations about the quality of applications that they develop. A user developer may be pleased with the quality of their "creation" and its contribution to their decision making activities when in fact the application includes serious errors such as incorrect formulae (Edberg & Bowman, 1996). End user developers who are unaware of quality problems in their applications may make errors in tasks or make poor decisions, which in turn could impact on organizational performance.

The potential for a user developer's perceptions to be colored by ignorance indicates the need for research assessing the ability of end users to evaluate the quality of the products of their own application development work. This can be accomplished by comparing user developers' perceptions of application quality with independent expert assessments.

The primary research question investigated in this study was:

How do user developer assessments of the quality of applications they have developed differ from independent expert assessments?

As discussed earlier, in previous studies that have related computing experience and training to EUC success (e.g., Al-Shawaf, 1993; Janvrin & Morrison, 1996; Raymond & Bergeron, 1992) the dependent variable used has mainly been user satisfaction and the results have not been conclusive. Hence in this study the second research question to be answered was:

How do experience and training influence differences between user developer and independent expert assessments of user developed applications?

It was hypothesized that:

- 1) End user assessments of UDA quality will not be consistent with expert assessments of UDA quality when the user developer has little experience with application development using the chosen tools.
- 2) End user assessments of UDA quality will not be consistent with expert assessments of UDA quality when the user developer has had little training in use of the chosen tools.

METHOD

The study was conducted with Masters of Business Administration (MBA) students participating in a business policy simulation over a period of 13 weeks as part of a capstone course in Strategic Management. All subjects had at least 2 years of previous professional employment.

The general applicability of research findings derived from student samples has been an issue of concern. However, Briggs et al. (1996) found MBA students to be good surrogates for executives in studies relating to the use and evaluation of technology, suggesting that the students who participated in this study can be considered as typical of professionals who would be involved in user development of applications in organizations.

The Game

The Business Policy Game (BPG) (Cotter & Fritzche, 1995) simulates the operations of a number of manufacturing companies. Participants assume the roles of managers, and make decisions in the areas of marketing, production, financing and strategic planning. Typical decisions to be made include product pricing, production scheduling and obtaining finance.

In this study the decisions required for the operation of each company were made by teams with 4 or 5 members. Decisions were recorded twice a week and the simulation run immediately afterwards so that results were available for teams to use during the next decision period. Each team was free to determine its management structure but in general the groups adopted a functional structure, with each member responsible for a different area of decision making. The simulation accounted for 50% of each subject's overall course grade.

The User Developed Applications

The subjects developed their own decision support systems using spreadsheets to help in their decision making. Decision support systems were developed either individually or by several members of a team. If they wished the subjects were able to use simple templates available with the game as a starting point for their applications, but they were not constrained with respect to what they devel-

oped, how they developed it, or the hardware and software tools they used. The majority of applications were developed in Microsoft Excel[®] but some subjects also used Lotus 1-2-3[®] and Claris Works[®]. The spreadsheets themselves were not part of the course assessment, so there were no formal requirements beyond students' own needs for the game.

Procedure for Data Collection

Each subject was asked to complete a written questionnaire and provide a copy of their spreadsheet on disk after eight "quarterly" decisions had been made (4 weeks after the start of the simulation). This point was chosen to allow sufficient time for the development and testing of the applications. The majority of completed questionnaires and spreadsheets were collected in person during the time when subjects were submitting their decisions but where this wasn't possible subjects were sent a follow up letter with a reply paid envelope. Ninety one questionnaires were distributed and 79 useable responses were received giving a response rate of 86.8%.

The Instrument

The questionnaire consisted of two sections. The first section asked questions about the subjects and their previous training and experience with spreadsheets, and the second section asked questions about the spreadsheet they had developed. Spreadsheet experience was measured in years and subjects were subsequently categorized (based on the spread of experience in the sample) as low experience (0 – 4 years experience), medium experience (5 – 8 years experience) or high experience (9+ years experience). Previous spreadsheet training was measured using a 4 item 5 point Likert-type scale from Igbaria (1990) which asked for level of training received in each of 4 types of training (college or university; vendor; in-company; self study). Scores for the 4 types of training were summed and subjects were subsequently categorized as low training (score less than 6), medium training (score of 7 - 9) or high training (score of 10 or more).

System quality relates to the quality of the IS itself and is concerned with matters such as whether or not there are "bugs" in the system, the consistency of the user interface and ease of use. In this study system quality was operationalized based upon the instrument developed by Rivard et al. to assess specifically the quality of user developed applications (Rivard et al., 1997). Rivard et al.'s instrument was designed to be suitable for end user developers to complete, yet to be sufficiently deep to capture their perceptions of components of quality.

Seven of the eight dimensions of quality in Rivard et al.'s instrument could be considered for these applications. These were reliability, effectiveness, portability, economy, user-friendliness, understandability, and maintainability. The verifiability

dimension was not included because the processes being examined in the questionnaire items relating to verifiability were not applicable to the environment in which the development was done. A number of individual items were also not included either because they were not appropriate for the applications under consideration (e.g., specific to database applications) or because they were not amenable to expert assessment (e.g., required either privileged information about the subjects' performance in the game or access to the hardware configurations on which the spreadsheets were originally used). Minor adaptations to wording were also made to reflect the terminology used in the BPG and the environment in which application development and use occurred.

The resulting system quality scale consisted of 40 items, each scored on a Likert scale of 1 to 7 where (1) was labeled "strongly agree" and (7) was labeled "strongly disagree." Measures for each of the quality dimensions were obtained by averaging the values of the criterion variables relating to that dimension. An overall application quality measure was obtained by averaging the seven quality dimension scores. This is consistent with the approach used by Rivard et al. The instrument had a Cronbach alpha of 0.82.

Independent Expert Assessment of System Quality

Two independent assessors using the same set of items also assessed the system quality of each UDA. Both assessors were information systems academics with substantial experience teaching spreadsheet design and development. Before assessing the study sample, the assessors completed four pilot evaluations to ensure consistency between the assessors. The ratings of the two independent assessors were generally very consistent. The sets of ratings for each application were compared, and were within 2 points of each other for the majority of items. Where scores for an item differed by more than 2 points the assessors re-examined the application together and reassessed their rating if appropriate.

RESULTS

Of the 79 subjects 78.5% were male and 21.5% female (62 males, 17 females). Their ages ranged from 21 to 49 with an average age of 31.8. Subjects reported an average of 5.9 years experience using spreadsheets (with a range from 0 to 15 years).

Table 1 indicates that the subjects had received relatively little spreadsheet training. More than 50% of the subjects had received no in-company or vendor training and just under 50% had received no college or university training. Self-study was the predominant means by which students had acquired their knowledge of spreadsheets.

Table 1: Summary of the subjects' previous spreadsheet training

Training Source	Level of Training										
	Mean	Number in each category									
		(1) None		(2)		(3)		(4)		(5) Extr. Intensive	
		N	%	N	%	N	%	N	%	N	%
College or University	2.0	46	58.2	8	10.1	6	7.6	11	13.9	7	8.9
Vendor	1.5	62	78.5	3	3.8	4	5.1	5	6.3	4	5.1
In-company	1.7	52	65.8	6	7.6	12	15.2	7	8.9	1	1.3
Self study	3.3	8	10.1	8	10.1	26	32.9	23	29.1	13	16.5

Table 2: A comparison of the mean user developer assessments of each quality dimension with the independent expert assessments for each quality dimension

Quality dimension	User developer assessment			Independent expert assessment			Significance
	Mean	Std. dev.	Ranking	Mean	Std. dev.	Ranking	
Economy	3.85	1.75	2	4.27	0.71	3	p=0.058
Effectiveness	3.77	1.29	5	4.29	1.03	2	p=0.009
Maintainability	3.56	1.44	6	3.29	1.25	4	p=0.228
Portability	3.91	1.31	1	4.51	0.68	1	p=0.001
Reliability	3.06	0.90	7	2.19	0.65	7	p=0.000
Understandability	3.83	0.83	3	3.20	0.71	5	p=0.000
User-friendliness	3.81	0.94	4	3.18	0.81	6	p=0.000
Overall quality	3.68	0.80		3.57	0.60		p=0.380

The first research question considered how end user developer assessments of application quality might differ from those of the independent experts. To address this question, the mean scores for each quality dimension as assessed by the user developers were compared with the independent assessments (Table 2). The scores for each quality dimension as assessed by the user developers were compared statistically with the independent assessments using paired samples t-tests.

There were significant differences on five of the quality dimensions. The user developers rated the effectiveness and portability of their applications significantly lower than did the independent assessors ($t=-2.67$, $p=0.009$; $t=-3.55$, $p=0.001$) and rated reliability, understandability and user-friendliness significantly higher than did the independent assessors ($t=7.25$, $p=0.000$; $t=4.58$, $p=0.000$; $t=4.06$, $p=0.000$). However, the overall assessments of quality were not found to be significantly different as the above differences canceled out. The rankings of mean quality across the dimensions were also considered. The applications were ranked highest on portability and lowest on reliability by both the user developers and the independent assessors, but the other dimensions were ranked differently.

Several individual questionnaire items stood out in illustrating problems that many end user developers had in recognizing quality problems with their applications. These are shown in Table 3. If end user developers have serious miscon-

Table 3: System quality instrument items on which there were major differences of opinion

	<i>% of applications for which end users developers agreed</i>	<i>% of applications for which expert assessors agreed</i>
Unauthorized users could not easily access all the data or a part of it	35.4	16.7
Each user owns a unique password	29.5	9.0
This system automatically corrects certain types of errors at data-entry time	35.0	0.0
This system always issues an error message when it detects an error	26.0	0.0
The system performs an automatic backup of the data	26.3	0.0
The system never modifies a cell without asking for a confirmation and getting a positive response	32.9	5.1

Table 4: A comparison of the assessments of each quality dimension across the low, medium and high experience groups

<i>Quality dimension</i>	<i>Low Experience</i>		<i>Med. Experience</i>		<i>High Experience</i>		<i>Significance</i>
	<i>Mean</i>	<i>Std. dev.</i>	<i>Mean</i>	<i>Std. dev.</i>	<i>Mean</i>	<i>Std. dev.</i>	
Economy							
End user developer	4.03	1.64	3.86	1.50	3.57	2.18	0.654
Expert assessors	4.24	0.73	4.16	0.77	4.48	0.58	0.294
Difference	-0.21	1.94	-0.30	1.80	-0.90	2.25	0.433
Effectiveness							
End user developer	4.07	1.19	3.41	1.32	3.85	1.27	0.141
Expert assessors	4.24	1.04	4.07	1.21	4.69	0.58	0.103
Difference	-0.17	1.64	-0.68	2.01	-0.82	1.40	0.367
Maintainability							
End user developer	3.75	1.55	3.86	1.09	2.88	1.52	0.037 ^{LH, MH}
Expert assessors	3.14	1.24	3.26	1.35	3.58	1.10	0.450
Difference	0.63	2.00	0.62	1.86	-0.70	1.53	0.022 ^{LH, MH}
Portability							
End user developer	4.02	1.43	3.79	1.09	3.83	1.49	0.797
Expert assessors	4.41	0.89	4.54	0.49	4.59	0.56	0.650
Difference	-0.40	1.66	-0.68	1.19	-0.76	1.62	0.652
Reliability							
End user developer	3.31	0.82	3.13	0.94	2.66	0.87	0.040 ^{LH}
Expert assessors	2.20	0.69	2.06	0.64	2.34	0.61	0.329
Difference	1.11	0.96	1.02	1.14	0.32	0.90	0.018 ^{LH, MH}
Understandability							
End user developer	4.16	0.69	3.80	0.74	3.45	0.98	0.011 ^{LH}
Expert assessors	3.18	0.68	3.12	0.83	3.37	0.61	0.476
Difference	1.02	1.06	0.66	1.23	0.08	1.23	0.026 ^{LH}
User-friendliness							
End user developer	3.95	1.05	3.92	0.71	3.47	1.00	0.145
Expert assessors	3.12	0.83	3.18	0.90	3.28	0.66	0.808
Difference	0.83	1.54	0.73	1.14	0.19	1.29	0.225
Overall quality							
End user developer	3.89	0.79	3.68	0.68	3.38	0.90	0.086 ^{LH}
Expert assessors	3.50	0.62	3.48	0.64	3.76	0.49	0.221
Difference	0.38	1.06	0.20	1.11	-0.38	0.99	0.043 ^{LH}

^{LH}Significant difference in means ($p < 0.05$) between the low experience and high experience groups

^{MH}Significant difference in means ($p < 0.05$) between the medium experience and high experience groups

ceptions such as these, it could pose significant risks to the security and integrity of organizational data and to the quality of organizational decision making.

The second research question considered whether experience and training might influence differences between user developer and independent expert assessments of user developed applications. The role of experience was considered first. End user developers were categorized according to the number of years of spreadsheet experience they had: low experience (0 – 4 years; $N = 29$), medium experience (5 – 8 years; $N = 29$) and high experience (9+ years; $N = 21$). Table 4 shows the mean quality assessments of the applications for the 3 groups of end user developers, the independent assessors and also the difference between the end user developer and independent assessment for each application. In order to analyze the differences in quality assessments between end users with different experience levels these were compared across the groups using ANOVA. In cases where ANOVA indicated significance differences the Bonferroni test was used to perform pairwise comparisons to determine the exact nature of the difference. The results provide support for Hypothesis 1.

The end user developers with low experience considered their applications to be of higher quality on all dimensions than did the end user developers with high experience. There were significant differences in end user quality assessments across the experience groups for maintainability ($F=3.45$, $p=0.037$), reliability ($F=3.36$, $p=0.040$) and understandability ($F=4.80$, $p=0.011$). In each of these cases Bonferroni tests showed that the quality assessments for the applications of the low experienced end users were significantly higher than those of the high experienced end user developers. However, no differences between the experience groupings were found in the independent assessments of quality.

A comparison of the difference scores between the 3 groups further supports what the results of the end user assessment comparison suggested. On each dimension and on overall quality the low experience group had either the largest positive difference or least negative difference. This pattern of differences suggests that the low experience group perceived applications of equivalent quality (as assessed independently) to be of better quality than did the high experience group. That is, they tended to overestimate the quality of their applications relative to the high experience group. The differences between groups were significant for overall quality ($F=3.27$, $p=0.043$) and for the following quality dimensions: maintainability ($F=4.02$, $p=0.022$), reliability ($F=4.24$, $p=0.018$) and understandability ($F=3.84$, $p=0.026$).

The role of training in the ability of an end user developer to make objective assessments of application quality was considered by comparing the 3 groupings of end user developers: low training (training score ≤ 6 ; $N=20$), medium training (score 7 – 9; $N=35$) and high training (score ≥ 10 ; $N=23$). Table 5 shows the

Table 5: A comparison of the assessments of each quality dimension across the low, medium and high training level groups

<i>Quality dimension</i>	<i>Low Training</i>		<i>Med. Training</i>		<i>High Training</i>		<i>Significance</i>
	<i>Mean</i>	<i>Std. dev</i>	<i>Mean</i>	<i>Std. dev</i>	<i>Mean</i>	<i>Std. dev</i>	
Economy							
End user developer	3.90	1.65	3.77	1.82	3.83	1.75	0.966
Expert assessors	4.32	0.67	4.20	0.75	4.32	0.70	0.761
Difference	-0.42	1.84	-0.43	2.16	-0.50	1.89	0.990
Effectiveness							
End user developer	3.75	1.33	3.62	1.28	3.96	1.26	0.624
Expert assessors	4.15	1.05	4.30	1.09	4.41	0.95	0.722
Difference	-0.40	1.81	-0.66	1.79	-0.45	1.66	0.844
Maintainability							
End user developer	3.74	1.22	3.27	1.56	3.78	1.38	0.330
Expert assessors	3.40	1.15	3.01	1.36	3.65	1.07	0.152
Difference	0.35	1.76	0.26	2.13	0.15	1.72	0.942
Portability							
End user developer	3.75	1.36	3.99	1.27	3.78	1.37	0.768
Expert assessors	4.31	1.04	4.58	0.55	4.57	0.41	0.338
Difference	-0.56	1.49	-0.59	1.56	-0.70	1.41	0.946
Reliability							
End user developer	3.16	0.93	2.82	0.70	3.33	1.08	0.097
Expert assessors	2.20	0.63	2.15	0.69	2.22	0.64	0.919
Difference	0.95	1.04	0.67	0.87	1.07	1.32	0.355
Understandability							
End user developer	4.07	0.69	3.78	0.75	3.70	1.03	0.337
Expert assessors	3.18	0.66	3.19	0.75	3.26	0.76	0.918
Difference	0.94	1.02	0.59	1.03	0.40	1.60	0.363
User-friendliness							
End user developer	3.93	0.80	3.74	0.93	3.79	1.10	0.762
Expert assessors	3.26	0.77	3.16	0.95	3.16	0.62	0.902
Difference	0.67	1.10	0.57	1.55	0.61	1.30	0.967
Overall quality							
End user developer	3.74	0.75	3.57	0.76	3.74	0.88	0.642
Expert assessors	3.55	0.63	3.51	0.63	3.66	0.56	0.688
Difference	0.19	1.05	0.05	1.13	0.08	1.13	0.902

mean quality assessments for the end user developers, the independent assessors and also the mean difference between the end user developer and independent assessment for each application. In order to analyze the differences in quality assessments between end users with different training levels these were compared across the groups using ANOVA. The results do not provide support for Hypothesis 2 as no significant differences were found between end users with low, medium and high levels of training with respect to end user developer quality ratings, independent quality ratings or difference scores on any of the quality dimensions. However it is interesting to note that the difference scores showed a similar (though not significant) pattern to the difference scores for the experience groupings with the low experience group having larger positive or less negative scores on all dimensions but reliability.

DISCUSSION

This study investigated the ability of end users to assess the quality of the applications they develop. The results indicate that there are some differences between the system quality assessments of end user developers and independent expert assessors, and also differences between quality assessments of end users with low and high levels of experience. In particular, the results of this study suggest that user developers with little experience may rate applications of equivalent quality more highly than do experienced user developers.

Can User Developers Assess the Quality of their Applications?

User developer assessments of overall application quality were not found to be significantly different from the independent assessments. This is because some of the differences at the quality dimension level are in different directions and partially cancel out. There were significant differences on four of the quality dimensions. The user developers rated the effectiveness and portability of their applications significantly lower than did the independent assessors. It is interesting that the user developers were more critical with respect to the effectiveness of applications than the independent assessors were. Of all the quality dimensions considered effectiveness is the dimension about which user developers should receive the most feedback via the BPG reports, and hence it is the dimension about which they could be expected to be most critical. The questionnaire items on portability related to two criteria: portability across hardware, and portability across organizational environments. User developer assessments differed significantly from the independent assessments only with respect to portability across different hardware platforms. This appears to result from a lack of awareness of just how portable applications developed in Microsoft Excel[®] currently are. The fact that both the end user developers and the independent assessors ranked portability highest amongst the dimensions suggests that the difference is not too problematic.

The user developers rated the reliability, understandability and user-friendliness of their applications significantly higher than did the independent assessors. Spreadsheets are the first introduction to application development for many end users, and in general end users have not been trained in systems analysis and design and tend to overlook issues such as reliability and auditability (Ronen, Palley, & Lucas, 1989). The differences in reliability and understandability assessments are consistent with the findings of Nelson (1991), who identified the major skill deficiencies of end users as being in technical and IS product areas, and with those of Edberg and Bowman (1996) who found major data integrity problems with the end user applications in their study. Rivard et al. (1997) noted that they would not be surprised to find user attitudes quite impervious to the more technical dimensions of application quality as the more 'technical' dimensions of quality

would be expected to preoccupy computer professionals but probably not end users unless they have been trained to focus on them. However the fact that reliability was the lowest ranking dimension for user developers as well as the independent assessors provides some hope that user developers are gaining insight into the weaknesses of their applications.

The difference in assessments of user-friendliness between the user developers and the independent assessors could be because the familiarity user developers gain with their applications during development may color their perceptions of their application's user-friendliness. As many UDAs are used by end users other than the developer (Bergeron & Berube, 1988) this could cause problems.

The Effect of Experience

Level of spreadsheet experience appeared to play an important role in the ability of end user developers to assess system quality. Those end users with little experience rated the quality of their applications higher on all dimensions than did the user developers in the high experience group. The differences between end user assessments of quality and independent assessments were also either larger (if positive) or less negative, for the low experience group. This suggests that lack of experience seriously impedes the ability of user developers to be objective about the quality of their applications. The quality dimensions for which the differences between experience levels were significant were the more technical dimensions of maintainability, reliability and understandability. It seems that despite Rivard et al.'s (1997) concerns about end user awareness of the technical dimensions of quality, with experience comes some increase in awareness.

It is interesting to note that no relationship was found between level of spreadsheet experience and the independent expert quality assessments. Those with more experience did not develop higher quality applications. Perhaps despite being more aware of the limitations of their applications they did not aim to develop quality applications. This could suggest a lack of awareness of the consequences of using applications of low quality (Ronen et al., 1989). A lack of concern for consequences might be exacerbated by two factors in this study. Firstly, the applications did not form part of the formal assessment for the course, and secondly, the subjects were aware that the applications would only be required for a limited period of time (the duration of the simulation). However these circumstances are often mirrored in the workplace with no external controls being placed on development and with end users developing applications that they believe will only be used once and then using them repeatedly (Kroenke, 1992). It can only be hoped that despite the fact that their applications were not of significantly better quality, the additional insight into the quality of their applications would lead high experience end users to treat their results with more caution.

The Effect of Training

In this study, level of spreadsheet training did not appear to play a role in determining either the ability of end user developers to assess system quality or system quality itself. The differences between the end user developer perceptions of quality and the independent assessments were not significantly lower for those end user developers in the highest training group. Both the amount of training that the subjects had received and the types of training could explain the results. As Table 1 shows, the subjects had received relatively little training and the major means of training was self-study. It has been suggested that when end users are self-taught the emphasis is predominantly on how to use the software rather than broader analysis and design considerations (Benham, Delaney, & Luzi, 1993). Thus the subjects in this study may not have received training of a type conducive to reflection on system quality. As self-training has been shown to be the major form of training in a number of studies (e.g., Benham et al., 1993; Chan & Storey, 1996) the results of this study may highlight potential problems in a wide range of organizations.

The fact that no relationship was found between amount of previous spreadsheet training and the independent quality assessments may also relate to the amounts and types of training received. Preliminary results of Babbitt, Galletta and Lopes's (1998) study of spreadsheet development by novice users suggested that end users whose training emphasizes planning and testing of spreadsheets will develop better quality spreadsheets. However it is also possible that despite the training the subjects may previously have had they did not consider it important to develop applications of high quality. Future research should investigate the role of type of training in both application quality and end user perceptions of application quality.

CONCLUSION

The results of this study cast some doubts on the ability of end users to make realistic determinations of the quality of applications they develop. Those subjects with little experience erroneously considered their applications to be of higher quality than subjects with more experience did. This may compromise the effectiveness of end users as application developers and could have major consequences when the systems developed are used to support decision making in organizations. Also of concern is the fact that no relationship was found between spreadsheet experience or training and the independent assessments of quality. Those user developers who would be expected to be more realistic in assessing the quality of their applications were, however, not developing applications of higher quality.

Given the increasing importance of user developed applications to organizational decision making it is essential that organizations be aware of the potential problems and that steps are taken to address them. Organizations must recognize

that end user developers may perceive the information from an application to be suitable to support decision making when, in fact, technical design and implementation flaws have introduced serious errors. With the majority of organizations imposing no quality control procedures on user developers (Panko & Halverson, 1996), training is perhaps the most effective tool for minimizing risks associated with end user computing (Cragg & King, 1993; Edberg & Bowman, 1996; Nelson, 1991). However, training must also emphasize application development methods and procedures, especially in the area of quality assurance, so that end users not only acquire the skills necessary to develop quality applications but also realize the consequences of not using these procedures. Unless user developer proficiency in developing applications is increased, organizations risk incurring considerable costs.

REFERENCES

- Al-Shawaf, A.-R. H. (1993). *An Investigation of the Design Process for End-user Developed Systems: An Exploratory Field Study*. Unpublished Ph.D., Virginia Commonwealth University.
- Amoroso, D. L. (1986). *Effectiveness of End-user Developed Applications in Organizations: An Empirical Investigation*. Unpublished Ph.D., University of Georgia.
- Babbitt, T. G., Galletta, D. F., & Lopes, A. B. (1998). Influencing the Success of Spreadsheet Development by Novice Users. *Proceedings of the Nineteenth International Conference on Information Systems*, 319-324.
- Benham, H., Delaney, M., & Luzi, A. (1993). Structured techniques for successful end user spreadsheets. *Journal of End User Computing*(Spring), 18-25.
- Bergeron, F., & Berube, C. (1988). The management of the end-user environment: An empirical investigation. *Information and Management*, 14, 107-113.
- Brancheau, J. C., & Brown, C. V. (1993). The management of end-user computing: Status and directions. *ACM Computing Surveys*, 25(4), 450-482.
- Briggs, R. O., Balthazard, P. A., & Dennis, A. R. (1996). Graduate business students as surrogates for executives in the evaluation of technology. *Journal of End User Computing*, 8(4), 11-17.
- Chan, Y. E., & Storey, V. C. (1996). The use of spreadsheets in organizations: Determinants and consequences. *Information and Management*, 31, 119-134.
- Cotter, R. V., & Fritzche, D. J. (1995). *The Business Policy Game*. Englewood Cliffs, NJ: Prentice-Hall.
- Cragg, P. G., & King, M. (1993). Spreadsheet modelling abuse: An opportunity for OR? *Journal of the Operational Research Society*, 44(8), 743-752.
- Crawford, J. B. (1986). *An Investigation of Strategies for Supporting and Controlling User Development of Computer Applications*. Unpublished Ph.D., University of California, Irvine.

- Edberg, D. T., & Bowman, B. J. (1996). User-developed applications: An empirical study of application quality and developer productivity. *Journal of Management Information Systems*, 13(1), 167-185.
- Etezadi-Amoli, J., & Farhoomand, A. F. (1996). A structural model of end user computing satisfaction and user performance. *Information and Management*, 30, 65-73.
- Igbaria, M. (1990). End-user computing effectiveness: A structural equation model. *OMEGA*, 18(6), 637-652.
- Janvrin, D., & Morrison, J. (1996). Factors influencing risks and outcomes in end-user development. *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, 2, 346-355.
- Kroenke, D. (1992). *Management Information Systems*. Watsonville, CA: McGraw-Hill.
- Mahmood, M. A., & Burn, J. M. (1998). No Substitute for Experience: A Meta-Analysis of End User Satisfaction Studies. In M. Khosrow-Pour (Ed.), *Emerging Information Technologies 1998 IRMA International Conference* (pp. 363-369). Hershey, PA: Idea Group Publishing.
- McLean, E. R., Kappelman, L. A., & Thompson, J. P. (1993). Converging end-user and corporate computing. *Communications of the ACM*, 36(12), 79-92.
- Nelson, R. R. (1991). Educational needs as perceived by IS and end-user personnel: A survey of knowledge and skill requirements. *MIS Quarterly*, 15(4), 503-525.
- Nelson, R. R., & Cheney, P. H. (1987). Training end users: An exploratory study. *MIS Quarterly*, 11, 547-559.
- Panko, R. R., & Halverson, R. P. (1996). Spreadsheets on trial: A survey of research on spreadsheet risks. *Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences*, 2, 326-335.
- Raymond, L., & Bergeron, F. (1992). Personal DSS success in small enterprises. *Information and Management*, 22, 301-308.
- Rivard, S., Poirier, G., Raymond, L., & Bergeron, F. (1997). Development of a measure to assess the quality of user-developed applications. *The DATA BASE for Advances in Information Systems*, 28(3), 44-58.
- Ronen, B., Palley, M. A., & Lucas, H. C. (1989). Spreadsheet analysis and design. *Communications of the ACM*, 32(1), 84-93.
- Shayo, C., Guthrie, R., & Igbaria, M. (1999). Exploring the measurement of end user computing success. *Journal of End User Computing*, 11(1), 5-14.
- Yaverbaum, G. J., & Nosek, J. (1992). Effects of information system education and training on user satisfaction. *Information and Management*, 22, 217-225.

Chapter 18

Toward an Understanding of the Behavioral Intention to Use a Groupware Application

Yining Chen and Hao Lou

Department of Accountancy and Department of Management Information
Systems

College of Business
Ohio University, Athens

INTRODUCTION

Over the past decade, groupware technologies, such as e-mail, electronic bulletin boards, and group support systems, have become an important part of the business computing infrastructure in many organizations. Organizations adopt groupware applications to enhance communication and collaboration among group members and thus improve group performance. While some groupware applications, e.g., e-mail, have been commonly accepted, many other applications, especially those that require significant collaboration and cooperation among users, are not widely used in organizations and their potential benefits are far from being fully realized (Orlikowski, 1993). Although numerous laboratory and field studies have consistently shown the relevance and positive impact of group support systems on group work, more research is needed in understanding how to increase the rate of diffusion and adoption of the technology (Nunamaker, 1997).

Behavioral-related elements, recognized by many, are the primary cause of resistance of users toward a newly implemented system or technology. Information technology (IT) research, however, tends to under-utilize existing knowledge in the behavioral science (Turner, 1982; Robey, 1979). Expectancy theory has been recognized as one of the most promising conceptualizations of individual motivation (Ferris, 1977). Many researchers have proposed that expectancy theory

can provide an appropriate theoretical framework for research that examines a user's acceptance of and intent to use a system (DeSanctis, 1983). This study uses expectancy theory as part of a student-based experiment to examine users' behavioral intention (motivation) to utilize a groupware application.

THEORETICAL BACKGROUND AND SUPPORTING LITERATURE

Groupware Acceptance and the Critical Mass Effect

Groupware refers to a class of computer technologies designed to support communication, collaboration, and cooperation among a group of knowledge workers. It covers a variety of technologies, ranging from simple e-mail systems to complex workflow applications. Although the use of some groupware technologies, such as e-mail, has become ubiquitous, organizations have encountered many difficulties in adopting and utilizing more sophisticated groupware applications, such as group support systems and Lotus Notes (Nunamaker, 1997; Orlikowski, 1993).

Prior Implementation Research

Prior implementation research indicates that user attitude toward the changes introduced by a system are thought to be especially important to the successful implementation of MIS applications (Barki and Huff, 1985; Ginzberg, 1980; Maish, 1979; Robey, 1979). This would indicate that measuring user attitude toward a system is essential for assessing system implementation success. Turner (1982) stressed that a continuing gap exists between the capabilities provided by new information systems and the extent to which these systems are accepted and used by individuals. This gap can be better explained by behavior-related elements than by elements strictly related to technical system attributes. Although behavioral-related elements are seen as the primary cause of resistance of users toward implementation of systems, implementation research has made little use of behavioral theory. Robey (1979) argued that "research in this area tends to underutilize existing knowledge in the behavioral science and typically fails to tie implementation research to more general models of work behavioral" (p. 528).

Expectancy Theory

Expectancy theory is considered one of the most promising conceptualizations of individual motivation. It was originally developed by Vroom (1964) and has served as a theoretical foundation for a large body of studies in psychology, organizational behavior, and management accounting (Harrell et al., 1985; Brownell and McInnes, 1986; Snead and Harrell, 1995; Geiger and Cooper, 1996). Expectancy models are cognitive explanations of human behavior that cast a person

as an active, thinking, predicting creature in his/her environment. He or she continuously evaluates the outcomes of his or her behavior and subjectively assesses the likelihood that each of his or her possible actions will lead to various outcomes. The choice of the amount of effort he or she exerts is based on a systematic analysis of (1) the values of the rewards from these outcomes, (2) the likelihood that rewards will result from these outcomes, and (3) the likelihood of reaching these outcomes through his or her actions and efforts.

According to Vroom, expectancy theory is comprised of two related models: the valence model and the force model. In our application of the theory, each user first uses the valence model and then the force model. In the valence model, each participant in a groupware application evaluates the system's outcomes (e.g., enhanced communication, increased ability to coordinate, better collaboration, and improved competence) and subjectively assesses the likelihood that these outcomes will occur. Next, by placing his/her own intrinsic values (or weights) on the various outcomes, each user evaluates the overall attractiveness of the groupware application. Finally, the user uses the force model to determine the amount of effort he or she is willing to exert to use the system. This effort level is determined by the product of the attractiveness generated by the valence model (above) and the likelihood that his/her effort will result in a successful contribution to the system. Based on this systematic analysis, the user will determine how much effort he/she would like to exert in participating in the groupware application.

Research Objectives

The general research question examined by this study is "Can the valence and force models of expectancy theory explain the motivation of a user to utilize a groupware application?" Specifically, under the valence model, we investigate the impact of the potential uses of groupware applications upon students' motivation to utilize such systems. The four uses of groupware applications tested by this study are (1) enhancing the communications among coworkers, (2) increasing the ability to coordinate activities, (3) gaining a better collaboration among coworkers, and (4) improving the competence of job performance. Under the force model, we examine the extent that the difficulty of using a groupware application will affect users' motivation to utilize the system. Based on the above research objectives, two research propositions are developed:

Proposition 1: The valence model can explain a user's perception of the attractiveness of using a new groupware application.

Proposition 2: The force model can explain a user's motivation to use a new groupware application.

RESEARCH METHOD

Subjects

The subjects were 86 undergraduate students¹ enrolled in five business courses taught by three different professors at a middle sized (15,000 to 20,000 total enrollment), mid-west university. Most of them had a junior or senior rank with a mean age of 21.5. The number of female and male are 44 and 42 respectively and 38 of them had used a groupware application in a prior course or other occasions.

Judgment Exercise

This study incorporates a well-established within-person methodology originally developed by Stahl and Harrell (1981) and later proven to be valid by other studies in various circumstances (e.g., Snead and Harrell, 1995; Geiger and Cooper, 1996). This methodology uses a judgment modeling decision exercise that provides a set of cues which an individual uses in arriving at a particular judgment or decision. Multiple sets of these cues are presented and each representing a unique combination of strengths or values associated with the cues. A separate judgment is required from the individual for each unique combination of cues presented.

We employed a one-half fractional factorial design² using the four second-level outcomes shown prior to Decision A. This resulted in eight different combinations of the second-level outcomes ($2^4 \times 2 = 8$ combinations). Each of the resulting eight combinations were then presented at two levels (10% and 90%) of expectancy to obtain 16 unique cases (8 combinations \times 2 levels of expectancy = 16 cases). This furnished each participant with multiple cases which, in turn, provided multiple measures of each individual's behavioral intentions under varied circumstances³. This is a prerequisite for the within-person application of expectancy theory (Snead and Harrell, 1995).

In each of the 16 cases, the participants were asked to make two decisions. The first decision, Decision A, represented the overall attractiveness of using the groupware application, given the likelihood (10% or 90%) that the four second level outcomes would result from their usage. (The instructions and a sample case are provided in Appendix I.) As mentioned earlier, the four second level outcomes are (1) enhancing communications among coworkers (2) increasing ability to coordinate activities (3) gaining better collaboration among coworkers, and (4) improving competence in job performance. The second decision, Decision B, reflected the strength of a participant's motivation to use the groupware application, using (1) the attractiveness of the system obtained from Decision A and (2) the expectancy (10% or 90%) that if the participant exerted a great deal of effort, he or she would be successful in using the system. We used an eleven-point response scale with a range of -5 to 5 for Decision A and 0 to 10 for Decision B.

*Table 1: Valence Model Regression Results **

	n	Mean	Standard Deviation	Range	Frequency of Significance at .05 Level
Adjusted R²	86	.6876	.2034	-.0267 to .9388	79/86
Standardized Beta Weight					
V1	86	.3748	.1745	-.4423 to .7646	62/86
V2	86	.3320	.1619	-.1506 to .6129	53/86
V3	86	.3190	.1830	-.5897 to .6803	51/86
V4	86	.5197	.2444	-.3965 to .9197	73/86

* Results (i.e. mean, standard deviation, range, and frequency of significant at .05) of individual within-person regression models are reported in this table.

V1: valence of communication enhanced
V2: valence of coordination ability increased
V3: valence of collaboration improvement
V4: valence of competence improvement

*Table 2: Force Model Regression Results **

	n	Mean	Standard Deviation	Range	Frequency of Significance at .05 Level
Adjusted R²	86	.7205	.2301	-.1141 to .999975	75/86
Standardized Beta Weight					
B1	86	.5997	.2530	-.1960 to 1.00	72/86
B2	86	.4976	.3110	-.2302 to .9763	64/86

* Results (i.e. mean, standard deviation, range, and frequency of significant at .05) of individual within-person regression models are reported in this table.

B1: weight placed on attractiveness of the groupware application
B2: weight placed on the expectancy of successfully using the system

Negative five represented “very unattractive” for Decision A and positive five represented “very attractive.” For Decision B, zero represented “zero effort” and ten represented a “great deal of effort.”

RESULTS

Valence Model

The first proposition predicts that the valence model of expectancy theory can explain a user’s perception of the attractiveness of using a groupware application. Through the use of multiple regression analysis, we sought to determine each participant’s perception of the attractiveness of participating in the evaluation. Decision A served as the dependent variable, and the four second-level outcome instruments served as the independent variables. The resulting standardized regression coefficients represent the relative importance (attractiveness) of each of the second-level outcomes to each participant in arriving at Decision A. The mean

adjusted- R^2 of the regressions and the mean standardized betas of each outcome are presented in Table 1. Detailed regression results for each participant are not presented but they are available from the authors.

As indicated in Table 1, the mean R^2 of the individual regression models is .6876. The mean R^2 represents the percentage of total variation in response that is explained by the multiple regression. Thus, these relatively high mean R^2 s indicate that the valence model of expectancy theory explains much of the variation in users' perception of the attractiveness of using a groupware application. Among the 86 individual regression models, 79 are significant at .05 level. These results support the first proposition.

The standardized betas of V1, V2, V3, and V4 are significant, at the .05 level, for more than half of the individuals in both groups. This implies that all four of the secondary outcomes were important factors, to a majority of the individuals, in determining the attractiveness of a groupware application. Although all four factors were important, some factors were more important than others. It is the mean of these standardized betas which explains how participants, on average, assess the attractiveness of potential outcomes resulting from a groupware application. The participants, on average, placed the highest valence on the outcome V4. The strength of the other valences, in descending order, was V1, V2, and V3. These results imply that the participants believe that improving job competence (V4) is the most attractive outcome of a groupware application and that improving collaboration among coworkers (V3) is the least attractive outcome. In the middle is the enhanced communication (V1) and increased coordination ability (V2).

Force Model

The second proposition proposes that the force model can explain a user's motivation to use a newly implemented groupware application. We used multiple regression analysis to examine the force model (Decision B) in the experiment. The dependent variable is the individual's level of effort to participate in the groupware application. The two independent variables are (1) each participant's perception about the attractiveness of the system from Decision A, and (2) the expectancy information (10% or 90%) which is provided by the "Further Information" sentence of the test instrument (see Appendix I). The force model results are summarized in Table 2.

The mean R^2 s (.7205) supports the second proposition and indicates that the force model sufficiently explains the students' motivation of participating in the evaluation system. The mean standardized regression coefficient B1 indicates the impact of the overall attractiveness of the groupware application while B2 indicates the impact of the expectation that a certain level of effort leads to successful participation in the system. These results imply that both factors, the attractiveness

of the groupware application (B1) and the likelihood that the user's efforts will lead to success (B2), are of similar importance to the user's motivation.

DISCUSSION AND IMPLICATIONS

This study provides a successfully illustration of expectancy theory, using the case of a groupware application. In practical terms, this study shows that expectancy can be applied early in the design phase of system development to provide a better indication of a user's intention to use a groupware application. In order to maximize system success (e.g., system usage and user acceptance), system analysts and designers may incorporate and stress the favorable attributes (second-level outcomes) identified in the study into their groupware application. Further, system developers may gauge their own effort to achieve these outcomes according to each outcome's relative importance as generated from the study.

Our empirical results show that the users have strong preferences for the uses of a groupware application and these preferences are remarkably consistent across individuals. To users, the most attractive outcome of a groupware application is the improvement of their job competence while the enhancement of communications among coworkers is the second strongest outcome. Thus, users who believe that their participation and use of the system will improve their competence or enhance communications should be highly motivated to participate in using the system.

Towards the goal of motivating users to participate in a groupware application, we make the following practical suggestions. First, declare prominently the uses and benefits of the groupware application in the users' training session, forums, and instruction manus. If these uses are consistent with the uses that users prefer and they believe that the system will truly be used for these purposes, the users will assign a high valence to the groupware application. The next step is to show users that their efforts in using the system can actually lead to the perceived benefits. Accomplishing this will increase users' subjective probabilities of the secondary outcomes. It would also increase their subjective probabilities that they will be successful in using the system. Thus, their force or motivation to participate will be high. One way of showing users that the system has been used successfully is to ask users to share on newsletter or users' meeting some recent examples of how the groupware application has helped accomplish a particular task or has helped the user improve his/her in job performance. This seems like a low cost, but highly visible way to show users the benefits of the system. It may also have the salutary effect of encouraging users to ponder and evaluate the benefits of the system, which in turn reinforces their opinion about the technology and reaffirms their acceptance decisions.

FOOTNOTES

- ¹ This study adopts a within-person methodology which does not have sample size requirement for making statistical inference. Prior studies (e.g., Burton et al., 1993; Geiger & Cooper, 1996), however, had sample size between 80 and 100.
- ² According to Montgomery (1984, p. 325), “if the experimenter can reasonably assume that certain high-order interactions are negligible, then information on main effects and low-order interactions may be obtained by running only a fraction of the complete factorial experiment.” A one-half fraction of the 2^4 design can be found in Montgomery (pp. 331-334). Prior expectancy theory studies (e.g., Burton et al., 1992 and Snead and Harrell, 1995) also used one-half fractional factorial design.
- ³ In the pilot test, we tested two different instruments; each had the order of the cases determined at random. The two instruments were distributed to every other student. We compared the average R^2 s from the two random order versions and found no significant difference between them. This result implies that there is no order effect in our experimental design.

REFERENCES

- Ashton, R.H., & Kramer, S.S. (1980). Students as surrogates in behavioral accounting research: some evidence. *Journal of Accounting Research*. 18(1), 1-15.
- Barki, H., & Huff, S.L. (1985). Changes, attitude to change, and decision support system success. *Information and Management*. 9(12), 261-268.
- Brownell, P., & McInnes, M. (1986). Budgetary participation, motivation, and managerial performance. *Accounting Review*. 61(4), 587-600.
- Burton, G.F., Chen, Y., Grover V., and Stewart, K.A. (1992). An application of expectancy theory for assessing user motivation to utilize an expert system. *Journal of Management Information Systems*. 9(3), 183-198.
- DeSanctis, G. (1983). Expectancy theory as explanation of voluntary use of a decision support system. *Psychological Reports*. 52(1), 247-260.
- Geiger, M.A., & Cooper, E.A. (1996). Using expectancy theory to assess student motivation. *Issues in Accounting Education*. 11(1), 113-129.
- Ginzberg, M.J. (1980). An organizational contingencies view of accounting and information systems implementation. *Accounting, Organizations, and Society*. 5(4), 369-382.
- Harrell, A.M., Caldwell, C., & Doty, E. (1985). Within-person expectancy theory predictions of accounting students' motivation to achieve academic success. *Accounting Review*. 60(4), 724-735.

- Harrell, A.M., & Stahl, M.J. (1984). Modeling managers' effort-level decisions for a within-persons examination of expectancy theory in a budget setting. *Decision Sciences*. 15(1), 52-73.
- Lucus, H.C. Jr., & Spitler, V.K. (1999). Technology use and performance: a field study of broker workstations. *Decision Sciences*. 30(2), 291-311.
- Maish, A.M. (1979). A user's behavior toward his MIS. *MIS Quarterly*. 3(1), 39-52.
- Montgomery, D.C. (1984). *Design and Analysis of Experiments*. New York: John Wiley & Sons.
- Murray, D., & Frazier, K.B. (1986). A within-subjects test of expectancy theory in a public accounting environment. *Journal of Accounting Research*. 24(2), 400-404.
- Nunamaker, J.F. Jr. (1997). Future research in group support systems: needs, some questions and possible directions. *International Journal of Human-Computer Studies*. 47, 357-385.
- Orlikowski, W.J. (1993). Learning from notes: organizational issues in groupware implementation. *Information Society*. 9(3), 237-250.
- Robey, D. (1979). User attitudes and management information system use. *Academy of Management Journal*. 22(3), 527-538.
- Snead, K.C., & Harrell, A.M. (1995). An application of expectancy theory to explain a manager's intention to use a decision support system. *Decision Sciences*. 25(4), 499-513.
- Stahl, M.J., & Harrell, A.M. (1981). Modeling effort decisions with behavioral decision theory: toward an individual differences model of expectancy theory. *Organizational Behavior and Human Performance*. 27(3), 303-325.
- Szajna, B., & Scamell, R.W. (1993). The effects of information system user expectations on their performance and perceptions. *MIS Quarterly*. 17(4), 493-516.
- Turner, J.A. (1982). Observations on the use of behavioral models in information systems research and practice. *Information and Management*. 5(6), 207-213.
- Vroom, V.C. (1964). *Work and Motivation*. New York: John Wiley & Sons.

APPENDIX I

INSTRUCTIONS

Assuming that you are employed by a company and consistently involved in group projects and assignments. A groupware application (e.g., Domino Discussion or Lotus Notes) is introduced to you and is available for your use. Various outcomes may result from using the application, such as: enhancing communications with your colleague; coordinating job-related activities; facilitating collabo-

ration among coworkers; and increasing competence in performing your job. Use of this application is voluntary; your use could range from minimum to maximum. Minimum use essentially implies that you will continue to perform your job as you have been without Lotus Notes. Maximum use means that you will rely on the groupware application to a great extent in performing your job.

This exercise presents 16 situations. Each situation is different with respect to how the groupware application is likely to be used. We want to know how attractive using the groupware application is to you in each given situation.

You are asked to make two decisions. You must first decide how *attractive* it would be for you to use the groupware application (DECISION A). Next you must decide how much *effort* to exert in using the groupware application (DECISION B). Use the information provided in each situation to reach your decisions. There are no “right” or “wrong” responses, so express your opinions freely. A sample situation is provided below. The 16 different situations start on the next page.

EXAMPLE QUESTIONNAIRE

If you use the groupware application (e.g., Domino Discussion or Lotus Notes) to the MAXIMUM extent in your job, the likelihood that:

You will enhance your communications with your coworkers isHIGH (90%)

You will improve your ability to coordinate job-related activities isHIGH (90%)

You will achieve a better collaboration among your coworkers isHIGH (90%)

You will increase your general level of competence in performing your job is ...LOW (10%)

DECISION A: With the above outcomes and associated likelihood levels in mind, indicate the *attractiveness* to you of using the groupware application in your job.

-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5	Very
											Attractive
Very Unattractive											

FURTHER INFORMATION: If you exert a great deal of effort to use Lotus Notes in your job, the likelihood that you will be successful in doing so isLOW (10%)

DECISION B: Keeping in mind your attractiveness decision (DECISION A) and the FURTHER INFORMATION, indicate the level of *effort* you would exert to use the groupware application.

0	1	2	3	4	5	6	7	8	9	10
Zero										Great Deal
Effort										of Effort

About the Editor

Dr. Tonya B. Barrier received her doctorate from the University of Texas at Arlington. Currently, she is an Associate Professor in the Computer Information Systems Department at Southwest Missouri State University. She specializes in end user computing, analysis and design, CASE tools and Human-Computer Interaction studies. She has articles published in journals such as the *Information Resources Management Journal* and *The Journal of Computer Information Systems*. She is on the editorial review board for the *Journal of End User Computing* and *Annals of Cases on Information Technology Applications and Management in Organizations*. She has been an active participant in the Information Systems Resources Management Association since 1991.

Index

A

a priori 94
 adaptive optimization technique 106
 Advanced GIS Systems 64
 aggregation 256
 annotations 256
 application domain 273
 articulatory distance 272
 automatic compatibility 163

B

B-Schema 194
 backtracking 259
 behavioral intention 304
 behavioral-related elements 304
 benchmarking 174, 178
 bilingual electronic catalog 18
 bottom-up approach 194
 browsing mode 20
 Business Policy Game (BPG) 292

C

canning 256
 canvas view (CV) 230
 centralized database system 97
 client and server 225
 CMU-WEB 219
 cognitive jump 236
 cognitive overhead 227
 coherence 227
 cohesion 237
 color scheme 66
 communication cost 104
 communication effectiveness 181
 communication interface 182

complete flexibility 225
 completeness 35, 36
 computer systems 5
 conceptual schema 199
 conditionalization 256
 consistency 272
 Constructivist 159
 content-dependent metadata 74
 content-level adaptation 256
 continuous improvement 174
 control 1
 CORBA (Common Object Request
 Brokered Architecture 221
 coverage 55
 critical data 64
 customer satisfaction variables (CSV) 174,
 177

D

data cleaning 277
 data collection 293
 data collection sheets 280
 data integrity 63
 data management 122
 data mining 184
 data warehousing 184
 data-driven (DD) approach 196
 database applications 194
 database designers 273
 database management system 61, 272
 database metadata management 86
 databases 259
 decision support system (DSS) 185
 decomposition 35, 36
 degree of support 220
 design environments 153

detective technique 97
 development management 129
 difficulty of navigation 228
 digital documents 72
 dimming 257
 direct guidance 259
 display scale 66
 Distributed Internet Databases 93
 document metadata management 84
 domain matter 47
 download time 237
 drill-down search 86
 dynamic anchor 77
 dynamic optimization 97
 Dynamic Systems Development Method 155

E

ecological data 274
 ecological sampling 275
 efficiency of use 229
 electronic catalog 18, 19
 electronic commerce (EC) 71
 elision 256
 end user computing (EUC) 1, 2, 54, 117, 134
 end user database development 271
 end user developers 34
 end user enhanceability 163
 end user expert systems 32, 43
 end user satisfaction 139
 end user systems 153, 162
 end user training 5
 end users 157, 271, 289
 enterprise-wide end user development 153
 environmental assessment 274
 environmental assessment database 280
 excessive use of color 251
 expectancy theory 305
 expert systems 33
 explicit annotations 258
 extended entity relationship(EER) 195
 Extensible Markup Language (XML) 222

F

feature density 63
 file management 54

filtering 256
 firm response variables (FRV) 174, 177
 focus group participants 247
 focus group questions 247
 focus group sessions 243
 focus groups 245
 footnote 256
 Force Model 309
 Format-Dependent Metadata 75
 functional integration 186

G

Geographic information systems (GISs) 53
 global coherence 227
 global electronic commerce 19
 global information management 19
 global users 18
 granularity 257
 group support systems 304
 groupware 305
 groupware application 304
 groupware technologies 304

H

heuristic evaluations 243, 245
 hiding 257
 hierarchical bookmark lists 259
 highlighting 257
 hybrid domain systems 74
 Hyper Text Markup Language (HTML) 222
 Hyperdocument Meta-information System (HyDoMiS) 71, 79
 hyperdocuments 78
 hyperlink outside application (HLOA) 231
 hyperlink to within application (HLWA) 231
 hyperlinks 72
 hypermedia document management 71
 hypermedia documents 71
 hypermedia information systems (HISs) 88
 hypertext applications 255
 hypertext transfer protocol (HTTP) 220

I

idempotent 224
 implicit reasoning 44

information chunk (IC) 230
 information chunk depiction (ICD) 230
 information design issues 250
 Information Needs 251
 information space 257
 information system (IS) 134, 137, 174, 261, 289
 information system quality 262
 information systems department (ISD) 134
 information systems design 174
 information systems use 141
 information technology 32
 Informix Universal Server (IUS) 110
 inspection methods 245
 interface representations 256
 Internet database systems 94
 interorganizational links 258
 interrelated events 220, 223, 225
 intertextual links 258
 intratextual links 258
 IS Component Technology 183
 IS Design 182
 isolated entities 210
 IT-oriented approach 3
 item analysis 24

J

join 55
 join queries 93, 98
 Joint Data- and Function-Driven (JDFD) approach 196
 judgment exercise 307

K

knowledge 31
 knowledge base 37
 knowledge base design 46
 knowledge capture 48
 knowledge management 46
 Knowledge-Based Systems 31

L

labeling map objects 65
 laissez-faire 119
 learnability 229
 link-level adaptation 257

load cost 105
 local coherence 227
 location of processing 220
 log-dependent metadata 75, 78
 low consistency 228

M

management support 9
 map objects 67
 mapping 272
 maps 66
 matching mode 21
 memorability 229
 meta-information system 71, 74
 meta-level adaptation 258
 metadata 60, 71, 72
 metadata classification 74
 metadata elements 75
 metadata management 82
 metadata schema 80
 microcomputer usage 142
 multidatabase systems 96
 Mutual Completeness Check 209

N

navigation 243
 navigation problems 249
 navigational system 255
 networked individuals 156
 node 76

O

Object Oriented Analysis and Design (OOAD) 226
 optical character recognition (OCR) technology 188
 organizational hyperdocuments (OHDs) 71
 organizational impact 143
 organizational integration 186
 organizational memory 73
 outdated and incomplete information 251
 outsource companies 135
 overhead due to consistency 236

P

paper-based media 255
 paradox of expertise 32
 parish records 274
 participating sites 94
 path analysis 24
 Phase II - 207
 pilot testing 247
 placement 257
 power distribution 13
 probabilistic reasoning 35, 36
 probe queries 100
 probe-based optimization technique 93
 problem statement 98
 prototyping 155

Q

quality measurement 35
 query sampling technique 97

R

Rapid Application Development 155
 relationship between information chunks
 (RIC) 231
 remote invocation cost 104
 Remote Method Invocation (RMI) 104
 remote server 113
 reporting 86
 resource allocation 128
 rule coupling 41
 rule length 40
 rule-based expert systems 33
 rules 272
 run-time behavior 108
 run-time optimization 95, 98, 100

S

scalability 102
 scrolling 250
 search engines 259
 search feature 250
 searching mode 21
 semantic distance 272
 semantic link labels 257
 semi-join plan 98
 servers 94

shopping cart 21, 24
 simple join plan 98
 site indexes 259
 site map 249, 259
 social event information 250
 software quality 35
 sorting 257
 sparse points 63
 specialized integrity checkers 65
 spiral model 155
 spreadsheet experience 300
 spreadsheet training 301
 Standard Generalized Markup Language
 (SGML) 222
 static anchor 77
 static query optimizer 94
 Static Query Optimizer (SQO) 99
 strategic integration 186
 stretchtext 257
 summarization 256
 supplemental navigation systems 258
 system implementation 81
 system quality 294
 system usability 261
 system-dependent metadata 75, 78
 systemic activity 154
 systems development life cycle 272
 systems development methodology 6

T

T-Schema 194
 tacit knowledge 44
 task mapping 182
 technological integration 186
 technology acceptability 9
 technology accessibility 9
 technology availability 9
 technology awareness 9
 technology development 10
 technology publicity 10
 technology spread 9
 termed cost-update points 106
 textual information 189
 theme 55
 time analysis 24
 timer 24
 top-down approach 194

Total Quality Management (TQM) 175
 TQM Communication 179
 TQM Implementation 174
 transfer protocol 220
 triangulation approach 243

X

X tuples 102

U

unified content model 20, 25
 universal polygons 63
 University Website 243
 usability 263
 usability experts 247
 usability practices 261
 usability testing 243, 246, 247, 263
 user access 64
 User Centred Design (UCD) 261, 264
 user developed applications (UDAs) 289, 292
 user developer assessments 299
 user interface adjustment 236
 user satisfaction 140
 user support 59
 user testing 245
 user-developed applications (UDAs) 271, 272
 user-interface adjustment 228
 user-oriented approach 5
 user-oriented control 1
 users' natural representations 271

V

Valence Model 308
 vendor off-the-shelf applications (vendor success) 135

W

Web applications 219
 Web page designers 250
 Web representation 255
 Web strategies 125
 Web-based development 121
 Web-based technologies 120
 Website Effectiveness Study 244
 workflow metadata management 82
 workflow-dependent metadata 75
 world wide web (WWW) 219

A New Title from IGP!



Business to Business Electronic Commerce: Challenges & Solutions

*Merrill Warkentin
Mississippi State University, USA*

In the mid-1990s, the widespread adoption of the Web browser led to a rapid commercialization of the Internet. Initial success stories were reported from companies that learned how to create an effective direct marketing channel, selling tangible products to consumers directly over the World Wide Web. By the end of the 1990s, the next revolution began—business-to-business electronic commerce.

Business to Business Electronic Commerce: Challenges and Solutions will provide researchers and practitioners with a source of knowledge related to this emerging area of business.

ISBN 1-930708-09-2 (h/c); US\$89.95; eISBN 1-591400-09-0 ;
308 pages • Copyright © 2002

Recommend IGP books to your library!



IDEA GROUP PUBLISHING

Hershey • London • Melbourne • Singapore • Beijing

1331 E. Chocolate Avenue, Hershey, PA 17033-1117 USA
Tel: (800) 345-4332 • Fax: (717) 533-8661 • cust@idea-group.com

See the complete catalog of IGP publications at <http://www.idea-group.com>



NEW from Idea Group Publishing

- **Data Mining: A Heuristic Approach**, Hussein Aly Abbass, Ruhul Amin Sarker & Charles S. Newton
ISBN: 1-930708-25-4 / eISBN: 1-59140-011-2 / 310 pages / US\$89.95 / © 2002
- **Managing Information Technology in Small Business: Challenges and Solutions**, Stephen Burgess
ISBN: 1-930708-35-1 / eISBN: 1-59140-012-0 / 367 pages / US\$74.95 / © 2002
- **Managing Web Usage in the Workplace: A Social, Ethical and Legal Perspective**, Murugan Anandarajan & Claire A. Simmers
ISBN: 1-930708-18-1 / eISBN: 1-59140-003-1 / 386 pages / US\$74.95 / © 2002
- **Challenges of Information Technology Education in the 21st Century**, Eli Cohen
ISBN: 1-930708-34-3 / eISBN: 1-59140-023-6 / 290 pages / US\$74.95 / © 2002
- **Social Responsibility in the Information Age: Issues and Controversies**, Gurpreet Dhillon
ISBN: 1-930708-11-4 / eISBN: 1-59140-008-2 / 282 pages / US\$74.95 / © 2002
- **Database Integrity: Challenges and Solutions**, Jorge H. Doorn and Laura Rivero
ISBN: 1-930708-38-6 / eISBN: 1-59140-024-4 / 300 pages / US\$74.95 / © 2002
- **Managing Virtual Web Organizations in the 21st Century: Issues and Challenges**, Ulrich Franke
ISBN: 1-930708-24-6 / eISBN: 1-59140-016-3 / 368 pages / US\$74.95 / © 2002
- **Managing Business with Electronic Commerce: Issues and Trends**, Aryya Gangopadhyay
ISBN: 1-930708-12-2 / eISBN: 1-59140-007-4 / 272 pages / US\$74.95 / © 2002
- **Electronic Government: Design, Applications and Management**, Åke Grönlund
ISBN: 1-930708-19-X / eISBN: 1-59140-002-3 / 388 pages / US\$74.95 / © 2002
- **Knowledge Media in Health Care: Opportunities and Challenges**, Rolf Grutter
ISBN: 1-930708-13-0 / eISBN: 1-59140-006-6 / 296 pages / US\$74.95 / © 2002
- **Internet Management Issues: A Global Perspective**, John D. Haynes
ISBN: 1-930708-21-1 / eISBN: 1-59140-015-5 / 352 pages / US\$74.95 / © 2002
- **Enterprise Resource Planning: Global Opportunities and Challenges**, Liaquat Hossain, Jon David Patrick & M. A. Rashid
ISBN: 1-930708-36-X / eISBN: 1-59140-025-2 / 300 pages / US\$89.95 / © 2002
- **The Design and Management of Effective Distance Learning Programs**, Richard Discenza, Caroline Howard, & Karen Schenk
ISBN: 1-930708-20-3 / eISBN: 1-59140-001-5 / 312 pages / US\$74.95 / © 2002
- **Multirate Systems: Design and Applications**, Gordana Jovanovic-Dolecek
ISBN: 1-930708-30-0 / eISBN: 1-59140-019-8 / 322 pages / US\$74.95 / © 2002
- **Managing IT/Community Partnerships in the 21st Century**, Jonathan Lazar
ISBN: 1-930708-33-5 / eISBN: 1-59140-022-8 / 295 pages / US\$89.95 / © 2002
- **Multimedia Networking: Technology, Management and Applications**, Syed Mahbubur Rahman
ISBN: 1-930708-14-9 / eISBN: 1-59140-005-8 / 498 pages / US\$89.95 / © 2002
- **Cases on Worldwide E-Commerce: Theory in Action**, Mahesh Raisinghani
ISBN: 1-930708-27-0 / eISBN: 1-59140-013-9 / 276 pages / US\$74.95 / © 2002
- **Designing Instruction for Technology-Enhanced Learning**, Patricia L. Rogers
ISBN: 1-930708-28-9 / eISBN: 1-59140-014-7 / 286 pages / US\$74.95 / © 2002
- **Heuristic and Optimization for Knowledge Discovery**, Ruhul Amin Sarker, Hussein Aly Abbass & Charles Newton
ISBN: 1-930708-26-2 / eISBN: 1-59140-017-1 / 296 pages / US\$89.95 / © 2002
- **Distributed Multimedia Databases: Techniques and Applications**, Timothy K. Shih
ISBN: 1-930708-29-7 / eISBN: 1-59140-018-X / 384 pages / US\$74.95 / © 2002
- **Neural Networks in Business: Techniques and Applications**, Kate Smith and Jatinder Gupta
ISBN: 1-930708-31-9 / eISBN: 1-59140-020-1 / 272 pages / US\$89.95 / © 2002
- **Managing the Human Side of Information Technology: Challenges and Solutions**, Edward Szewczak & Coral Snodgrass
ISBN: 1-930708-32-7 / eISBN: 1-59140-021-X / 364 pages / US\$89.95 / © 2002
- **Cases on Global IT Applications and Management: Successes and Pitfalls**, Felix B. Tan
ISBN: 1-930708-16-5 / eISBN: 1-59140-000-7 / 300 pages / US\$74.95 / © 2002
- **Enterprise Networking: Multilayer Switching and Applications**, Vasilis Theoharakis & Dimitrios Serpanos
ISBN: 1-930708-17-3 / eISBN: 1-59140-004-X / 282 pages / US\$89.95 / © 2002
- **Measuring the Value of Information Technology**, Han T. M. van der Zee
ISBN: 1-930708-08-4 / eISBN: 1-59140-010-4 / 224 pages / US\$74.95 / © 2002
- **Business to Business Electronic Commerce: Challenges and Solutions**, Merrill Warkentin
ISBN: 1-930708-09-2 / eISBN: 1-59140-009-0 / 308 pages / US\$89.95 / © 2002

Excellent additions to your institution's library! Recommend these titles to your Librarian!

To receive a copy of the Idea Group Publishing catalog, please contact (toll free) 1/800-345-4332, fax 1/717-533-8661, or visit the IGP Online Bookstore at: [\[http://www.idea-group.com\]](http://www.idea-group.com)!

Note: All IGP books are also available as ebooks on netlibrary.com as well as other ebook sources. Contact Ms. Carrie Stull at [\[cstull@idea-group.com\]](mailto:cstull@idea-group.com) to receive a complete list of sources where you can obtain ebook information or IGP titles.

Series

in Information Technology Management

- ✓ *Advanced Topics in Database Research Series*
- ✓ *Advanced Topics in Global Information Management Series*
- ✓ *Advanced Topics in End User Computing Series*
- ✓ *Advanced Topics in Information Resources Management Series*
- ✓ *Cases on Information Technology Series*

Expand your library collection in IT by ordering these cutting-edge publications today!

Add these IGI Series to your personal or library IT collection today. Each series will greatly enhance your collection in information technology. You will benefit from these publications on advanced research in various areas of information technology innovation, applications, utilization, management and organizational and societal issues.

Cases on Information Technology Series (ISSN 1537-9337)

<u>Vol</u>	<u>Copyright</u>	<u>ISBN</u>	<u>Price</u>	<u>Qty</u>
4-1	2002	1-930708-40-8	US\$89.00	___
4-2	2002	1-930708-16-5	US\$74.95	___
4-3	2002	1-930708-27-0	US\$74.95	___
3-1	2001	1-878289-61-6	US\$89.00	___
2-1	2000	1-878289-83-7	US\$89.00	___
1-1	1999	1-878289-56-X	US\$89.00	___

Advanced Topics in Database Research Series (ISSN 1537-9299)

<u>Vol</u>	<u>Copyright</u>	<u>ISBN</u>	<u>Price</u>	<u>Qty</u>
1-1	2002	1-930708-41-6	US\$74.95	___

Advanced Topics in Information Resources Management Series (ISSN 1537-9329)

<u>Vol</u>	<u>Copyright</u>	<u>ISBN</u>	<u>Price</u>	<u>Qty</u>
1-1	2002	1-930708-44-0	US\$74.95	___

Advanced Topics in Global Information Management Series (ISSN 1537-9302)

<u>Vol</u>	<u>Copyright</u>	<u>ISBN</u>	<u>Price</u>	<u>Qty</u>
1-1	2002	1-930708-43-2	US\$74.95	___

Advanced Topics in End User Computing Series (ISSN 1537-9310)

<u>Vol</u>	<u>Copyright</u>	<u>ISBN</u>	<u>Price</u>	<u>Qty</u>
1-1	2002	1-930708-42-4	US\$74.95	___

► **RECOMMEND THIS IT SERIES TO YOUR LIBRARY.**



IDEA GROUP PUBLISHING

Hershey • London • Melbourne • Singapore • Beijing

1331 E. Chocolate Avenue, Hershey, PA 17033-1117 USA

Tel: (800) 345-4332 • Fax: (717) 533-8661 • cust@idea-group.com

