

AMERITECH

**STANDARDS FOR
INFORMATION SYSTEMS**

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Ameritech Graphical User Interface Standards and Design Guidelines

© Copyright Ameritech, Inc., 1992, 1993, 1994, 1995, 1996. All rights reserved.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Written by

Robert M. Schumacher, Jr.
Ameritech Services, Inc.
Technology Management

Reviewed¹ by

Michael Pickard, Ameritech Services, Inc.
Arnold Lund, Ameritech Services
Eileen Schwab, Ameritech Services, Inc.
Ray Bennett, Ameritech Services, Inc.
Rick Omanson, Ameritech Services, Inc.
Jeffrey Multach, Ameritech Services, Inc.
Stacey Harris, Ameritech Services, Inc.
Mary Hardzinski, Ameritech Services, Inc.
Jane Kerjes, Ameritech Services, Inc.
Rich Lankford, Ameritech Services, Inc.
Lloyd Swager, Ameritech Services, Inc.
Scott Beckwith, Ameritech Services, Inc.
Brian Kowalski, Ameritech Information Systems, Inc.
Steve Steele, Ameritech Information Systems, Inc.
Michael Muller, Bellcore
Bob Root, Bellcore
Jayson Webb, User-Centered Technologies, Inc.
George Tortolero, CompuPros, Inc.

Trademarks

OSF/Motif and Motif are trademarks of the Open Software Foundation, Inc.

IBM is a registered trademark and OS/2, Common User Access, CUA, Systems Application Architecture and SAA are trademarks or service marks of International Business Machines Corporation.

Microsoft is a registered trademark and Software Development Kit, SDK, Visual Basic, Windows, Excel, and Word are trademarks of Microsoft Corporation.

Apple and Macintosh are registered trademarks of Apple Computer, Inc.

UNIX and Open Look are registered trademarks of AT&T.

OSCA is a trademark of Bell Communications Research.

¹ Many thanks to the reviewers for their hard work and constructive criticism. They are not responsible for any errors, misconceptions, or vagaries. The standards and guidelines in this document do not necessarily represent the positions of all of the reviewers.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Table of Contents

Ameritech Graphical User Interface Standards and Design Guidelines....	1
Table of Contents.....	5
Preface	9
1. Introduction.....	10
1.1. Purpose of standards and guidelines	10
1.1.1. Benefits for users	11
1.1.2. Benefits for designers.....	11
1.1.3. Benefits for Ameritech.....	11
1.2. Using this document	12
1.2.1. Scope	12
1.2.2. Audience.....	12
1.2.3. Relationship of this design guide to	12
platform style guides.....	12
1.2.4. Assumptions	13
1.2.5. Document organization	13
1.2.6. Document conventions.....	15
2. User-Centered Design.....	17
2.1. A user-centered design	18
2.1.1. Early and continual focus on users	18
2.1.2. Integrated design	19
2.1.3. Early and continual user testing.....	19
2.1.4. Iterative design	20
2.2. Design heuristics	21
2.2.1. Simplicity	21
2.2.2. Conceptual consistency.....	22
2.2.3. Design for error	24
2.2.4. Provide good feedback	24
2.2.5. Support different interaction styles	25
2.2.6. Avoid the use of modes.....	26
2.3. User interface architectures	27
2.4. Fundamentals of object-based interaction.....	28
2.4.1. Objects	28
2.4.2. Actions and Operations	29
2.4.3. Object-action model	32
2.4.4. Basic and expert techniques.....	33
2.4.5. Basic techniques	33
2.4.6. Expert techniques	33
2.4.7. General guidelines for use.....	34
2.5. Interaction styles.....	35
2.5.1. Indirect manipulation.....	35
2.5.2. Direct Manipulation	35
2.6. The desktop metaphor	36

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3. The Ameritech Standards Reference Guide: Combined Standards for Macintosh, Windows, and OSF/Motif Applications.....	38
3.1. Interface objects	39
3.1.1. Windows	39
3.1.2. Dialog boxes	56
3.1.3. Menus	71
3.1.4. Other standard controls	80
3.2. Navigation	84
3.2.1. Cursor navigation	84
3.2.2. Keyboard navigation	86
3.2.3. Focus	87
3.2.4. Selection	87
3.3. Other topics	89
3.3.1. Use of color	89
3.3.2. Use of sound	90
3.4. The Ameritech Signature	92
3.4.1. Use of logo	92
3.4.2. Ameritech standard menus	92
3.4.3. Ameritech standard dialog boxes	107
3.4.4. Ameritech color palettes	111
3.4.5. Common interface objects	111
4. Guidelines for Obtaining Input from the User	113
4.1. Selecting from a Set	114
4.1.1. "One of many" Choices	114
4.1.2. "N of many" Choices	133
4.2. Issuing commands	141
4.2.1. Labeling Commands	142
4.2.2. Navigation	143
4.2.3. Buttons	144
4.2.4. Menus	146
4.2.5. Icons	149
4.3. Getting field input	151
4.3.1. Elements of entry fields	151
4.3.2. Placement of display elements	158
4.3.3. Navigation	163
4.3.4. Edit checks	164
4.3.5. Data forms	165
4.4. Putting it all together: Organizing controls	167
4.4.1. Good practices of display layout	167
4.4.2. Navigation	173
5. Guidelines for Providing Output to the User	174
5.1. General	175
5.1.1. Display output	175
5.1.2. Avoiding visual clutter	176
5.1.3. Data groupings	176

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.1.4.	Abbreviations	177
5.1.5.	Highlighting	177
5.1.6.	Labels	182
5.1.7.	Fonts.....	182
5.1.8.	Searching text or lists.....	184
5.2.	Information displays.....	185
5.2.1.	Tables.....	185
5.2.2.	Data records	188
5.2.3.	Text	190
5.2.4.	Flowcharts.....	193
5.3.	User guidance	195
5.3.1.	General.....	195
5.3.2.	Handling user errors.....	196
5.3.3.	Messages.....	198
5.3.4.	On-line help	200
5.3.5.	Feedback	202
6.	Porting to GUI Platforms.....	205
6.1.	GUI to GUI	205
6.2.	Non-GUI to GUI.....	205
7.	References and index	206
7.1.	References	206
7.2.	Index	208

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Preface

The purposes of the Ameritech Graphical Look and Feel Standards (*aka* design guide) are twofold. First, it provides a basis for a consistent user interface look and feel across Ameritech applications and platforms. Second, it is the initial attempt to integrate research on user interface design directly into the selection and usage of interface objects found in user interface libraries and toolkits.

It is also necessary to say what the design guide is *not*. I have attempted to stay away from duplicating, in any major way, things that can be found readily in existing style guides, except where there are differences from the existing standard. I do, however, refer to the style guides where necessary. In short, the design guide is meant to complement existing style guides by narrowing them to specific Ameritech purposes and giving the reader more guidance in selecting objects. Only at points of conflict, which are explicitly indicated in the text, do the Ameritech standards take precedence.

This standard is still evolving. Because the Windows user interface style guide was released shortly before completion of this document, it is possible that not all Windows standards are properly recorded. Windows standards were assumed through study of existing products and using the closely related IBM SAA CUA standard.

The design guide has seven sections:

- Section 1 - Introductory material on standards and guidelines
- Section 2 - User-centered design and architecture
- Section 3 - Ameritech standards
- Section 4 - Obtaining input from users
- Section 5 - Presenting output to users
- Section 6 - Portability issues from text to GUI and GUI to GUI
- Section 7 - Supporting materials (appendices, index, etc.).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1. Introduction

1.1. Purpose of standards and guidelines

The Ameritech Graphical User Interface Standards and Design Guideline (also referred to as the “design guide”) provides a common reference for development of Ameritech products using a graphical user interface to products.

How is this document different from a style guide, such as OSF/Motif™ Style Guide 1.1 or IBM CUA Advanced Design Reference? Most style guides are interface object catalogs. They provide a description of the object and tell something about how it works. However, style guides offer the reader little in the way of good design practices and when certain objects should and should not be used. Furthermore, style guides rarely discuss how to use combinations of different types of objects. Style guides are still necessary, but they do not in themselves mean that good, usable interfaces will necessarily be developed (i.e., they are under-specified).

There is a body of knowledge offering advice on developing good, usable application interfaces (specifically, the field of human-computer interaction). Unfortunately, in the day-to-day work of application development, that information is difficult to find, read, and apply, and thus it is often ignored.

That’s where the goals of this design guide come into play. Simply, the main goal in writing this design guide is to bridge the gap between the traditional style guides and the human-computer interaction literature. This fusing of approaches will result in better applications being developed for Ameritech. To the extent that good user interface practices work their way into choices developers make, the primary goal has been accomplished. Moreover, in the course of writing this document I came to realize that there are many places where “macro” user interface objects (such as standard dialog boxes) could be constructed and become candidates for re-usability. The attempt here is to take some of the uncertainty out of development, and in the process improve application usability and corporate consistency.

The use of look and feel standards through a design guide and platform style guides have many benefits. Those are covered next.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1.1.1. Benefits for users

Standards and guidelines provide a consistency that users appreciate and to which they grow accustomed. Consistency for users translates into faster learning, greater efficiency, fewer errors, and increased satisfaction. In short, adherence to the guidelines presented here should result in improved application usability. Standards also allow relatively easy movement across platforms.

1.1.2. Benefits for designers and developers

This design guide also provides benefits to application designers and developers. Some development decisions are handled more straightforwardly in this design guide than with current style guides. Furthermore, the long-term goal is to make many of the interface objects discussed here available through the Ameritech re-use library, thus minimizing coding from scratch. As you develop the graphical user interface (GUI) code, it too could be added to the re-use library.

This document also contains examples and descriptions of good practices. These good practices are things that you might consider when writing your application, making it more readily acceptable to users.

1.1.3. Benefits for Ameritech

Ameritech benefits through reduced costs if applications are more usable. Saving ten seconds on a transaction may not seem like much, but across 75 transactions per day with 5000 service representatives, it is a very significant savings to Ameritech. Also the image of the company internally and externally is better when products look and feel the same. Finally, with code being re-used the productivity of developers increases as well.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1.2. Using this document

1.2.1. Scope

This design guide is intended to provide guidance to the majority of decisions that developers have to make about graphical user interfaces (GUIs). If the information the developer needs is not in this document there are pointers to many other supporting documents that should be at the developers disposal. Also, there are areas where this document lacks specificity (e.g., on help). Some parts of this document will not be completed until later. Comments on this document are encouraged, and can be made by contacting the author.

1.2.2. Audience

The intended audience for this document is anyone who is responsible, in any way, for the look and feel of an application. Developers will be its primary users. Product managers may be interested in order to be sure that the products with which they are involved conform to what is accepted standard. This document will also be helpful for anyone who monitors products delivered to Ameritech having a GUI component.

1.2.3. Relationship of this design guide to platform style guides

This document does *not* replace platform style guides. In writing this, I had no interest in duplicating or restating what could be found in the style guides except where the practices in this document differed from those in the style guides.

At the end of important sections a heading, "More reading", is presented. The intent of "More reading" is to provide you with direct access to the relevant sections of the style guides and other supporting documents. *You should always consult the platform style guide in addition to this document.* In some cases, for example toggled menu items in Section 4.1.1.1.5, the platform style guide has little or nothing to say about the preceding topic. In that case, you should follow the advice of this style guide. You are also highly encouraged to read the sections of the other references cited under "More reading." These references are for other platform style guides and secondary references. Often the treatment of issues is stronger in one style guide or reference than in another, and looking at the other style guides and readings will give you a greater appreciation for the issues involved.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Because this document is an amalgam of three style guides you will find that some references to the style guides are not as pertinent to the just-presented material as others. If, for example, the information here is more specific than that covered in the style guide, you *should* follow the advice presented here. However, if the information here is less specific than the style guide, you should follow the style guide. To put it another way, when in doubt adopt the more specific approach.

1.2.4. Assumptions

This design guide makes the following assumptions:

- products will be developed on either Microsoft Windows 3.x capable machines (e.g., IBM PS/2 Model 50), Apple Macintoshes, and/or UNIX workstations running X Windows with OSF/Motif.
- developers will be creating graphical user interfaces using either the Microsoft Software Development Kit, Apple toolbox, or OSF/Motif.
- the hardware and software platform supports multitasking in a windowed environment and a bit-mapped display.
- each user will have a keyboard and a pointing device, most likely a mouse.

1.2.5. Document organization

1.2.5.1. Chapter 2 - User-centered design

Chapter 2 provides valuable background material on questions to ask and things to consider when developing an application. Applications are tools: the GUI is how that tool appears to the user. There is a separation between how the application is constructed at the task analytic level and how that application appears to the user. You might have the most conforming, best looking GUI ever developed, but if the task that underlies that GUI is not well thought out the GUI cannot make up for it. Chapter 2 gives you a user-centered perspective on how to design an application and how the application design fits with the development of the GUI.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1.2.5.2. Chapter 3 - The Ameritech standards reference guide: Combined standards for Macintosh, Windows, and OSF/Motif applications

Chapter 3 is the most important of all the chapters in the design guide. Chapter 3 lays out the required elements for a GUI, such as menu items, window types, standard dialog boxes, etc. This chapter lays out the shell -- the starting point -- for most applications.

1.2.5.3. Chapters 4 & 5 - Guidelines for obtaining input from the user & guidelines for providing output to the user

As noted above, standards and style guides only take the developer so far. They prescribe the proper use of objects. Where style guides fall short is ensuring that the proper objects were selected for the application in the first place. Chapters 4 and 5 are presented in order to help you make good selections for the conditions you might encounter when designing an application. They are broadly arranged by input and output: how to get information from the user and how to display information for the user.

A second reason for the development guide is that there is a fairly broad and well-researched body of literature on user interface design. This information has been collected and (hopefully) made accessible to you in a useful and usable way. This information is termed "good practices" rather than "standards" because, outside of the context of the application, it is impossible to state precisely the correct form for an item.

While these sections do not contain issues regarding precise conformance to the Ameritech standard, they should be used heavily.

1.2.5.4. Chapter 6 - Porting user interfaces across platforms

To be completed in a future release.

1.2.5.5. Chapter 7 - Reference Information

This chapter provides supporting information: index, references, cross-references between terminology, etc.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1.2.6. Document conventions

1.2.6.1. Requirements conventions

There are two key levels at which this document is written. First, some items are “required.” The required items are the standards -- there is strong enough evidence to say that a certain way of doing things must be standard practice. Most of these requirements are not hard to achieve (e.g., “every window must have a border”). A “√” indicates a requirement. The term “must” is typically used for required elements. Requirements are also surrounded by a box, e.g.,

√ Menu items that have cascaded menus must not have accelerators.

The other level is one of “guidelines” -- there is enough information to make sound recommendation as to the desired practice that, without good and defensible reason, should not be violated. A “+” indicates a guideline. The term “should” is typically used for guidelines, e.g., “+ For column orientation place the label above the text boxes, with the label flush-left with the left edge of the column of text boxes.”

There is another level: good practices. Good practices usually refer to situations where the advice is less specific or highly conditionalized. A “Δ” indicates a good practice. For example, “Δ Design the application so as to minimize the number of times the user has to travel between the keyboard and the mouse. If the user is constantly going back and forth, it greatly diminishes performance.”

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

1.2.6.2. Text conventions

COURIER	Courier font indicates a menu item or command button label in the text.
FULL CAPS	Full caps indicate something the user would type, e.g., CHICAGO.
SMALL CAPS	Small caps indicate names of keys found on most keyboards, e.g., SHIFT.
Bold	Bold indicates that an important term is used for the first time. When a term is defined, it is surrounded by a box, e. g.,

Secondary windows are a type of window that supplement the interaction in the primary window.

1.2.6.3. Terminology conventions

Point	Position the pointer (by moving a pointing device -- typically a mouse) on a screen object.
Click	Press and release a pointing device button without moving the pointing device.
Double-Click	Press and release the pointing device button twice in rapid succession without moving the pointing device.
Drag	Press and hold the pointing device button while moving the pointing device.
Choose	Select a menu command or a command button.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

2. User-Centered Design²

Good interface design follows from good design practice and from understanding the principles that underlie human-computer interaction. Design principles are the fundamentals that guide the development of specific standards and guidelines. This chapter discusses the basic principles from which many of the design guidelines that follow are derived. The literature in human-computer interaction is rich with principles of user interface design. The discussion presented here represents a sample of the literature on user-centered design.

2.1. A user-centered design philosophy

Applications are usually developed according to some philosophical reference point. This point of reference for development can be data-centered, technology-centered or user-centered. In any development these three points of view are highly interdependent; the difference is which one receives the most attention. The data-centered and technology-centered approaches are common -- they involve looking at the data flows and technology with the user considered an input/output device. Fundamentally, applications are tools used by users. Tool design is a science and an art, but the tool must be capable of being used in order to be fully effective. Effectiveness only comes from focus on the needs of the user. Thus, a user-centered design should be the starting point for applications. Before assuming that you know and practice user-centered design, read this section carefully and take the advice seriously. Good design is not an accident. In short, practicing user-centered design means spending two or three times longer listening to users than in a data-centered or technology-centered model, translating what they need into prototypes, establishing usability requirements, their reactions, and making modifications. This analysis time pays off quickly in greater certainty in the effectiveness of the product, reduced development times, and fewer modification requests.

The four steps in a user-centered approach are as follows: early and continual focus on users, integrated design, early and continual user testing, and iterative design (adapted from Gould, 1988). There is much

² Much of this section was taken (with permission) from GUI Design Guidelines for Bellcore Software Products, Issue 1.1. Many thanks for Bob Root at Bellcore for allowing me to use this.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

more to this than is presented here, but if you follow the four basic steps your application will be better. The skills involved in design and evaluation of software interfaces are specialized and are in the domain of human factors. Your application could benefit from consulting with an expert in human factors.

2.1.1. Early and continual focus on users

One of the keys to successful application development is to develop an early focus on the users of the application. Remember that the user's goal is not to use computers or to use the application; the goal is to accomplish some task -- establish a new account, find a bill payment center, or locate a wire pair. Make direct contact with users; listen to them.

The following questions will guide your understanding of the users and the tasks. As you uncover the answers to these questions let them guide how you think about design. Also, the answers to these questions help to focus the choices that you will make during design (e.g., in Chapter 4).

- Why does someone use the application?
- When do they use the application?
- What are the characteristics of users?
- Where will they use the application?
- What is the skill level of the users?
- How much exposure have they had to computers?
- How well do they know the tasks involved?
- How often will they have to use the application?
- How do users do the tasks now?
- How suited is the technology choice to the user?
- What task knowledge needs to be built into the application?

2.1.2. Integrated design

All aspects of the design should consider that quality, in the eyes of the customer, is paramount. That usually means that the user's needs for productivity and satisfaction are met. Thus, all aspects of design have to be considered as they relate to usability concerns.

2.1.3. Early and continual user testing

A developer would not think of releasing software that has not undergone thorough code review and considerable system testing.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

However, many developers skip or shortchange testing of the most critical system component: the user. User testing (often called “usability testing”) is not simply gathering opinions. The purpose of code reviews is to find flaws and weaknesses in code; the purpose of user testing is to find flaws and weaknesses in how the users interact with the application. User tests can be constructed that are every bit as quantifiable and rigorous as established methods for testing software quality.³ As a developer, you are *not* an impartial judge of the interface. You have different (probably much more technical) skills than the user community and you will use the application differently than your users. The only way to uncover and confront usability problems is through user testing.

User testing is often neglected because there is a lack of information on what it is, how to do it, and when to do it. In short, user testing involves having a sample of representative users perform typical tasks under realistic constraints. It should be done early enough in the process so that key problems can be identified and rectified before the software is released. Running usability tests requires some expertise -- contact someone in the human factors department for help. Usability tests can be conducted on all aspects of the user’s experience with the application -- GUI, documentation, training, etc.

A future release of this document will further specify the usability assurance process, although “look and feel” and usability are now becoming a part of the Ameritech Product Assurance process.

2.1.4. Iterative design

Once the design has been created and tested, use the output from the to make changes. The design-modify-test cycle is critical for getting the best application user interface. Use contact with users to find flaws in the application, not in the users. Continually try to understand what users need and want and be willing to make changes as a result. Remember, it’s impossible to get the design right the first time.

³ There is a temptation to declare, all too readily, that usability problems are documentation and training issues. Following this temptation often means that the application is doomed to failure *in the eyes of the user*. Documentation gathers dust and training is costly and soon forgotten. Usability problems are application design problems; they cannot and should not be relegated to the documentation or training.

2.2. Design heuristics

Along with a user-centered design philosophy, there are some basic rules of thumb, or heuristics, for designing a usable application (adapted from Nielsen & Molich, 1990; and Norman, 1988). Many of the required and recommended elements in the following chapters follow from these heuristics:

- Simplify the task and the interface as much as possible
 - Reduce visual noise
 - Adopt the user's familiar natural language
 - Minimize what the user has to remember
 - Do NOT rely on training and documentation to solve software problems
- Be conceptually consistent
 - Develop a clear conceptual model of the application
 - Do not violate users' expectations
- Design to prevent error
 - Constrain actions to the appropriate ones
 - When handling errors, provide good error messages
- Provide good feedback
 - Users should see what is happening
 - Make the available choices salient to the user
- Support different interaction styles
 - Provide many ways to select objects and actions
 - Provide shortcuts for expert users
 - Minimize the number of keystrokes
- Avoid the use of modes.

Discussion of these heuristics follows.

2.2.1. Simplicity

Simple interfaces are easier to work with, easier to learn, and less likely to create opportunities for error than complex ones. Simplicity is achieved by reducing complexity. Many of the design principles

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

discussed here contribute to reducing the apparent complexity of the interface. For example,

- Visual complexity can be reduced by putting only the most important controls in buttons or control panels and hiding the rest in menus. The menu controls are still available but they don't clutter up the display.
- Verbal complexity can be reduced by studying and employing the words and concepts that the users of the application use. Use words common in the user's vocabulary and *appropos* to the task. Positive, active words are most appropriate.
- Task complexity can be reduced by grouping related tasks together and by reducing the number of different tasks presented in a single window. User activities should be simple at any given moment, though they may be complex taken together. Furthermore, provide only a few alternatives at a time and provide only choices that are valid -- appropriate actions should be constrained. Provide clearly marked exits for the user.
- Conceptual complexity can be reduced by helping the user form a clear model of how the application is organized and how the different components can be accessed.
- Training is rarely complete and quickly forgotten. Documentation is rarely read and, when it is used, information is hard to find. Adopt the assumption that there is no training and no documentation available to the user.

Increasing the simplicity of the task or the interface should not be confused with simplifying the task so much that performance is actually impaired. The analyst/designer has a fine line to walk; there is a tradeoff between making the interface somewhat complex and requiring training, if as a result expert performance is well-supported.

2.2.2. Conceptual consistency

Conceptual consistency is the bedrock of good interface design. The interface should be based on a clear and consistent conceptual structure; that structure being revealed to the user through the design and behavior of the interface components. A well-structured interface organizes and helps structure the user's activities. It achieves this by presenting objects and actions appropriate to the tasks being performed and by indicating the flow of activities required to carry out a task. In some cases the interface may explicitly restrict the user to performing

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

certain tasks in a certain sequence. In other cases the design will permit the user to perform tasks in parallel or in a number of different sequences. One of the easiest ways to achieve good organization is to perform a thorough task analysis (see "A user-centered design philosophy" below) and use it to construct a task model for the interface.

Achieving consistency in GUI design is the *raison d'être* for this design guide. Within an application, consistent interface design reduces user errors and improves performance across a range of tasks. Consistent use of standard operations and components helps users predict the results of an action before they perform it, and helps them form a consistent mental model of how the interface works. Across applications, consistent interface design supports positive transfer of training and reduces the time required to learn a new application. Once users know how to perform a certain task or operation in one application, they should be able to perform a similar task in a different application without retraining. Some of the basic principles of consistent design are:

- Similar interface components should have similar appearance and behavior. For example, pulldown menus and popup menus should both look like menus and behave like menus, even though they may be accessed in very different ways.
- An action should always have the same result regardless of context (i.e., avoid modes). For example, pushing the up arrow in a scrollbar should always scroll the contents in the same direction, regardless of whether one is manipulating a scrolling list or an entire window.
- Standard functions should be reused across tasks and should be presented in the same way in each task. By definition, the behavior of standard functions should be consistent across different task contexts. For example, the standard cut, copy and paste functions should be provided in an Edit menu whenever those functions are needed in a task.

2.2.3. Design for error

The best cure for errors is prevention. However, errors will happen even in the best interfaces and with the most skilled and experienced users. The next best thing to prevention is minimizing the consequences of errors and helping the user to recover from them. The following techniques can be used to assist the user.

- Provide clear error messages. Error messages should state the problem in the user's terms, not the systems. Telling users that "Error -

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

16" occurred is not helpful. Telling them that the network is down is more informative. Error messages should also indicate appropriate actions for the user to take to recover from the error or correct the problem that caused it.

- Provide undo mechanisms that allow the user to reverse an unintended operation before proceeding to the next one.
- Force the use of explicit destruction when an operation can't be undone. Explicit destruction means asking the user to take an extra step to perform operations that result in permanent alterations to data or other important system assets. The most common technique is asking the user to confirm operations such as deleting data.

2.2.4. Provide good feedback

Feedback is the application's way of communicating with the user. Feedback from the interface helps users confirm that the application has received their input. It also informs the user about the status of a task and the state of objects in the interface. Feedback is conveyed primarily through change in the visible appearance of objects on the screen. For example, selecting an object may cause it to be highlighted. Feedback can also be conveyed through explicit mechanisms such as percent-done indicators telling the user about the status of a task in progress. In general, every user input to the application should be confirmed with visual feedback. Feedback may also be provided through other modalities such as sound, but visual feedback is the primary means for letting the application communicate with the user.

The data objects and operations provided by the application must be easily available understandable, and not require special documentation or formal training. Usually this means making them visible on the display as graphical objects. Visibility of data objects and operations supports learning by exploration and helps remind the user of potential operations that can be performed, and makes it easier to provide good feedback. Availability has other meanings as well. It can mean representing the *possibility* of an operation even though the operation itself is not currently visible. Menus, for example, hint at the presence of different categories of operations but don't reveal them until the user asks for them. Availability can mean that anything visible on the screen is available for use. If a find operation produces some output on the screen, say a text string, that output is available as input to another operation, perhaps by using the copy and paste operations. The availability principle can also be used to guide decisions about which operations should be visible and which may be concealed. Operations

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

that are central to task performance, frequently used, or are needed to remind the user about important activities have high availability requirements and should be presented in buttons or control panels for instant access. Operations that are secondary in importance or are infrequently accessed have low availability requirements and can be placed in menus or dialog boxes.

2.2.5. Support different interaction styles

There's usually more than one way to perform a task, and different users are likely to prefer different techniques. Flexibility means providing multiple input methods for performing an operation or accomplishing a task. A menu pick, for example, could be accomplished by pointing at it with the mouse, by pressing a function key, or by using arrow keys to traverse the menu structure. Activating a particular window in a complex application could be accomplished by selecting the window name from a menu, by clicking on a visible window with the mouse, or by typing the name of the window in a command dialog box. A Find operation could be performed in many different ways. In general, flexibility allows users of different skill and experience levels to perform tasks in the way that is best for them, consistent with the general principles of good design.

In practice, flexibility is achieved in two ways: by supporting multiple input techniques (keyboard and mouse, for example) and by providing expert techniques and accelerators for certain tasks. However, expert techniques should never replace the basic techniques. An interface should usually be designed with the relatively inexperienced user in mind. Expert techniques should be layered on top of the basics to improve the expert's efficiency but they should not interfere with ease of learning or simplicity of operation for the less expert user.

To facilitate learning, command key equivalents (accelerators) should be displayed next to their corresponding menu item selections. Likewise, when a command key is pressed, the menu on which that command resides should be momentarily highlighted.

2.2.6. Avoid the use of modes

Users enter a mode when identical commands or keystrokes have different actions in different situations. Modes are to be avoided because they force the user think about how the application works and not on the task she or he is doing. Modes cannot be completely avoided, and in some cases are even desired: modal dialog boxes, such

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

as ones informing the user of an impending destructive action, are necessary and appropriate. Always allow the user a clear way to remove herself or himself from the mode.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

2.3. User interface architectures

Properly done, user testing will expose design weaknesses that can cause usability problems. Unfortunately, these weaknesses are often difficult to modify because the software architecture of the application being written does not allow for easy modifiability of the user interface. More often than not, the code that drives the user interface is tangled with the application code, and changes in the look and feel of the interface require expensive and time-consuming changes to the application code.

The Bellcore OSCA architecture aids in the development of code that is functionally discrete, thereby disentangling the user interface code from applications code. The OSCA architecture is only one example of the general class of architectures that separate the user interface code from the applications code. In general, architecturally discrete user interface code is referred to as a user interface management system (UIMS).

The observance of strict UIMS-inspired software development rules allow relatively easy changes to the user interface. The data collected during user testing, new user interface technologies, new user groups with different skill levels, and expansion of system functionality can all be accommodated within the UIMS model. In fact, UIMS software engineering technology is the most cost-effective means by which a design philosophy based on early and continual user testing and iterative design can be realized.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

2.4. Fundamentals of object-based interaction

In a graphical user interface, both the data and the possible operations on the data are made visible to the user. Application data are represented as data *objects*. User activities, or operations, are represented as visible *controls*. Interaction is carried out by selecting objects and activating controls with a pointing device (usually a mouse). This is sometimes called a "point & click" style of interaction, in contrast with the "type at the system" style characteristic of command-oriented interfaces. A mouse is not required for a GUI interface -- the keyboard is also used for textual data entry and many operations are mapped onto keyboard equivalents and accelerators so that keyboard intensive tasks don't require user to switch between input devices.

The key element in GUI design is visibility -- making both the data and the associated operations visible to the user through the use of graphical display elements. But the GUI designer must be concerned both with graphical objects and with techniques of user input -- ways for the user to express intentions to the application. To summarize, the fundamental principles of GUI interface design are⁴:

- Continuous visible representation of objects and actions
- Physical actions (e.g., drag and drop) or concrete operations (e.g., labeled button presses) instead of typed-in command names
- Rapid, incremental, reversible operations

The guidelines that follow are designed to operationalize these principles and make them concrete.

2.4.1. Objects

is a general term for any of the visible elements of a GUI display.

⁴Shneiderman (1982, 1983); Ziegler and Fähnrich (1990).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Objects can be just about anything visible on the display, including icons, windows, buttons, menus, elements of interactive graphics, list items, individual characters of text or data, and labels. This style guide is concerned with three types of objects:

- user input controls
- data output objects
- data input objects.

Any other components described in the style guides that do not fall into these three categories will be referred to as “interface components”.

are objects that allow the user to perform operations on data objects and to manipulate the application and elements of the interface itself. Controls are activated by user actions. Controls include buttons, sliders, radio buttons, scroll bars, icons and menus.

represent visible elements of application data that do not accept direct user input. Icons depicting elements of an editable network are one example.

Data objects are defined by the application. Guidelines for designing data objects will be developed in a future issue of the style guide.

allow text or data to be entered into the application, e.g., text boxes. Data input objects may be populated in several ways, including keyboard input, application activities, or as a result of a cut and paste operation.

2.4.2. Actions and Operations

User input may be separated into two categories: actions and operations.

User are physical behaviors that the user performs with an input device, such as the mouse or the keyboard. Examples include mouse clicks and keypresses.

are the result of user actions on data objects or user interface controls. They represent the functions that allow users to manipulate the data objects, control the application, and manipulate the elements of the user interface. Examples include opening a file, finding a string in a list, or deleting an object by dragging it to a trashcan.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Actions are mediated by input devices. Feedback from actions is observed on the screen (e.g., movement of the mouse pointer reflects the movements of the mouse) but actions cannot be represented as concretely as visible controls. Actions are necessary for low level activities such as selecting objects and for direct-manipulation style interactions. Actions can only be *performed*, they cannot be activated or selected in the way that controls can be. The meaning of an action depends on the physical characteristics of the user's behavior (e.g., which mouse button is pressed), the characteristics of the object being manipulated, and the application context.

Operations take place when a user action (a mouse click or a keypress) activates a control. Operations are concretely represented in visible GUI controls and menus. The meaning of an operation is defined by the form, label and behavior of the control. Thus, scroll bars move window contents in a horizontal or vertical plane, while buttons perform the operation indicated by the button's label. Operations provide the basic mechanisms for manipulating data and controlling an application.

A special class of operation is found in direct manipulation style interactions, where operations are *expressed* through physical actions with the mouse but are not represented concretely in interface controls. Direct manipulation interactions have many non-standard characteristics. They will be discussed in detail later on.

2.4.2.1. Explicit and Implicit operations

The object-action model generally follows an **explicit operation model**. In the explicit operation model all available operations are represented visibly and concretely in the form of selectable user interface controls.

There is also an **implicit operation model**, in which the operation is expressed through a particular user action instead of a visible control.

Implicit models are necessary in direct manipulation style interfaces, where the semantics of a drag and drop action are typically dictated by the context -- dropping a file icon on the trash can implies a different operation from dropping it on a file folder. However, implicit models are also found in GUI interfaces based on selectable operations instead of drag and drop actions. Two examples: (1) a Find operation is performed by selecting a text box, typing in a string, and pressing the ENTER key on the keyboard. No Find control is visible anywhere in the interface; (2) a new record in a database application is created by entering a nonexistent string in a key field -- the application assumes that a new record is being created and clears the work area.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Implicit operations are invisible to users -- they are "hidden" in the input device and the behavior of the interface. Users are expected to know how perform the desired operation. If they don't know it, there is no way for them to discover it. If they are casual users of the application, there is no way to learn incrementally or to relearn things once they're partly forgotten. In addition, implicit operations are usually based on some set of assumptions about the task and the user that may not always be valid. Implicit operations violate one of the foundation principles of GUI design -- visibility of data objects and operations. For these reasons, implicit operations should be used only for direct manipulation interactions and to implement expert techniques that supplement basic techniques based on explicit operations.

2.4.2.2. Generic operations

Much of the power of the GUI model comes from the use of a few generic, or universal, operations that can be applied to nearly any class of object in many different contexts. Well-known examples include *Undo*, *Cut*, *Copy* and *Paste* from the Edit menu. The universality of the operations makes the interface easy to learn and easy to use across a broad range of tasks. In general, the use of generic operations provides the following benefits:

- improves the consistency and predictability of the interface: users always know how to delete or copy an object and where to find the operation.
- encourages the user to develop a consistent object-action model of task performance.
- reduces the size of the command set: the same operation may be used in many tasks
- increases the flexibility of the design: the same operations are always available regardless of changes in the object structure of the application

2.4.3. Object-action model

In the **object-action model** (also known as the select and operate paradigm) users first select an object and then perform some action on the selected object. The object-action model is contrasted with the **action-object model** whereby users first choose an action to perform then pick an object upon which the action should be performed.

Selection and operation are typically performed with the mouse, but the keyboard may be used as well. The application responds to each operation with explicit visible feedback. If the operation modifies an

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

object, the object's visible appearance changes to reflect its changed state. Tasks are accomplished by stringing together a sequence of select-and-operate steps, each of which may be "undone" before proceeding to the next step. This model allows both for selecting multiple objects for a single operation and for performing multiple successive operations on a single object. *The object-action model is the default for Ameritech software products.*

In practice, it is not always possible to apply the object-action model to every user interaction in an application. In some cases an action-object interaction is necessary or even desirable. Descriptions of action-object model and guidelines for choosing between action-object and object-action models need to be developed.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

2.4.4. Basic and expert techniques

There are generally several ways to implement a given operation or support a task. In the general case there are two distinct approaches to interaction design -- basic and expert techniques.

2.4.5. Basic techniques

Basic techniques follow from the principles of GUI design, the explicit operations model and the select-and-operate paradigm. They have the following characteristics:

- all data objects are visible, and all visible data objects can be manipulated
- all available operations are represented concretely in the form of buttons, menu items, or other graphical controls
- any operation can be accomplished by first selecting a data object, then activating an appropriate control
- selecting a data object constrains the availability of operations to those appropriate for the data. Unavailable operations are indicated by visual lowlighting.

Basic techniques support learning by exploration, easy relearning, consistency and predictability. They make it possible for casual users to successfully interact with a variety of applications. These benefits derive from the facts that (1) all operations are available for visual inspection, either by looking at the display or by browsing through the contents of menus with the mouse; (2) all operations can be performed using the mouse or keyboard to activate a visible control; (3) no operations are "hidden" in the keyboard or in a mouse button.

2.4.6. Expert techniques

Expert techniques are shortcuts for experienced users. They generally follow the implicit operations model, in which operations are hidden in the mouse or invisible objects such as popup menus. Expert techniques also take the form of accelerators, such as keystroke equivalents of menu picks or double clicking the mouse on certain data objects. They depend heavily on task and object context to help the user remember the technique and apply it appropriately. They have the following characteristics:

- usually invisible to the user; cannot be learned through exploration

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- usually idiosyncratic to a particular task, application, or data context
- usually involve specific user actions to invoke them

Expert techniques, when well designed, can increase efficiency of operation for expert users, give the user flexibility in task performance, and increase the throughput of repetitive operations. When based on a careful task analysis they can also improve the mapping between the user's conceptual model of the task and its implementation in a GUI.

2.4.7. General guidelines for use

Always design the interface around basic techniques first, then add the expert techniques as supplemental ways of doing things. The use of basic techniques is the key to consistency. Expert techniques cannot be expected to be consistent over tasks or applications.

2.5. Interaction styles

2.5.1. Indirect manipulation

Indirect manipulation is where the user specifies the object with one action (e.g., selecting text) and specifies the operation to be performed on the object by a separate action (e.g., Cut); it is based on the explicit operation model.

With indirect manipulation the user does not actually manipulate an object to perform an operation. Indirect manipulation is the most common GUI interaction style. This style of interaction is conceptually very close to a command-oriented interaction, except that the command is specified by pointing and clicking on visible objects rather than by typing strings.

2.5.2. Direct Manipulation

In **direct manipulation** (DM) interactions operations are expressed through physical actions performed with the mouse instead of labeled button presses or menu picks. The most common DM action is the **drag and drop**, in which the user selects an object in the usual way, then uses the mouse to drag the object to an appropriate destination and drop it there.

When DM is used, the operations are implied in the relationship between data objects, user actions, and the display context. Thus, the *meaning* of drag-and-drop (i.e., the semantics of the operation) is determined by the context in which the action takes place. For example, if a data object is dragged from one pane of a window to another, the action may be interpreted as a "copy-object" operation. If it is dragged to a trash can icon, the action may be interpreted as a "delete-object" operation.

Direct manipulation operations can be very complicated for users because, by their nature, they are not explicit or visible. However, if used in the proper context, DM can be powerful and effective. Details of DM operations will be defined in detail in a future release. The drag and drop interaction model is discussed in IBM SAA CUA AIDR p. 78-81, MSG §3.2.1, Windows ADG, p. 34-36, 39-40.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

2.6. The desktop metaphor

Graphical user interfaces allow objects to be represented similarly or analogously to familiar, real-world objects. Since people have experience with the familiar objects, computers applications can emulate their behaviors, resulting in applications that are intuitive and easy to learn. For example, just as tools like scissors and tape are available on your real desktop to support office tasks, the computer desktop offers utilities to cut and paste.

At first glance, the desktop metaphor may not appear well suited for the intended users of a particular application, but if all Ameritech applications are accessed via mechanisms using this metaphor, it will become easy and natural for users, and they can generalize their knowledge to all applications (including commercially available ones). The desktop metaphor must be used by all applications, and is discussed in greater detail below. The desktop metaphor is used at the top level of the application (i.e. to open it and to interact with the primary window), and it should be enhanced with application-specific actions or replaced with other appropriate real-world metaphors. After an application is opened, it should use a real-world metaphor appropriate to its functions. For example, in an inventory application, displays might reflect the actual equipment they represent, and equipment could be placed or removed from locations which look like shelves and slots.

Throughout the document, the term "desktop" refers to the metaphor for the "surface" the user sees on their screen. This is an analogy to the surface of a desk in the real world.

The desktop metaphor has some of the following basic characteristics: a file as a place to store related information; folders used to group related data files (as in a real office, existing files can be opened and new files can be created); multiple, overlapping windows representing stacks of paper which are commonly found on desk tops. The ability to edit information online has resulted in the creation of additional actions which have become a *de facto* standard, available in most commercial GUI tool kits. The most common of these include:

Save	Undo
Save as	Cut
Print	Copy
Exit/Quit	Paste

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3. The Ameritech Standards Reference Guide: Combined Standards for Macintosh, Windows, and OSF/Motif Applications

Contents

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1. Interface objects

3.1.1. Windows

A **window** provides context for an application; all interface objects are organized and presented in windows. For instance, in a word processing application the window provides a workspace for the text. A window is the principal way in which a user interacts with an application.

+ An application may place one or more windows on the display, each showing a different view or providing added information to the user.

3.1.1.1. Types

3.1.1.1.1. Primary windows

A **primary window** provides the location of the main user interaction with the application.

+ An application may have multiple primary windows. For instance, a spreadsheet application may have multiple spreadsheets open at the same time. Primary windows are independent of other primary windows in the application.

+ It is strongly recommended that every application initially display a primary window. Displaying a primary window helps set the application context for the user. Furthermore, that primary window should be displayed as soon as possible -- do not leave the screen blank while starting the application.

More reading

Apple HIG, pp. 42-48.
OSF/MSG, § 2.3.2.2, 5.2.
IBM SAA CUA AIDR, pp. 9, 186-187.
Windows ADG pp. 49-54, 211.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.1.1.2. Secondary windows

Secondary windows are a type of window that supplements the interaction in the primary window. Dialog boxes are a special type of secondary window dealing specifically with command completion and passing information to the user; they are covered in detail in Section 3.1.2.

√ If used, secondary windows must be related to a primary window.

+ When secondary windows are initially requested (e.g., user selects a command linked to a secondary window, such as `Open...`), they should be displayed (also called **posted**) so that they cover a portion of the active window. Also, if at all possible, they should not be posted on top of any part of the window that a user needs to see in order to complete her or his task.

+ Secondary windows may call other secondary windows to further supplement the interaction, if needed.

+ Secondary windows should be movable, if at all possible. Modeless windows should be movable, modal windows are usually not movable (see Section 3.1.1.1.2.1).

One of the main characteristics of secondary windows is whether they are modal or modeless.

3.1.1.1.2.1.

Modal secondary windows require the user to perform some action on the window before continuing to work on the application. For example, if the user wants to dial out on a modem, he or she chooses `Dial...` from a menu and then the application presents a modal secondary window requesting the user to enter a phone number. Secondary windows can be -- they require information from the user before continuing with the current application, but not from continuing on other applications. Or secondary windows can be -- they require information from the user before continuing with anything else.

√ When presenting a modal secondary window, users must take an explicit action to dismiss it, such as selecting `OK` or `Cancel`.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ Except in certain situations, use of application modal secondary windows is discouraged (see Sections 3.1.2.1.2, 3.1.2.1.3, and 3.1.2.1.4).

+ Use system modal secondary windows only for serious problems, like out of memory or bad media errors.

3.1.1.1.2.2.

A **modeless secondary window** allows users to perform other operations without dismissing the secondary window.

√ When a floating palette or toolbox is presented as a modeless secondary window it must remain completely visible (i.e., it must remain at a layer above the window to which it applies).

+ Modeless secondary windows are recommended for dialogs that don't require immediate attention and for commands that do not need to be completed before continuing.

More reading

Apple HIG, p. 42-48.
OSF/MSG, § 2.3.2.2, 5.2.
IBM SAA CUA AIDR, p. 10, 221-222.
Windows ADG, p. 49-54, 125-127.

3.1.1.2. Required window elements

Figure 3-1 shows examples of the four required elements of a window: border, client area, window menu (for Windows and Motif) and title.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

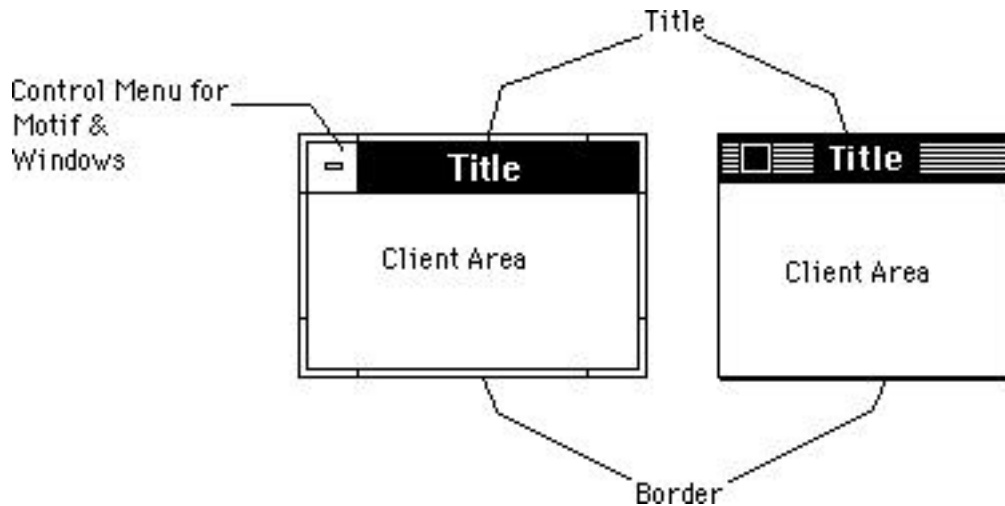


Figure 3-1. Required window elements for Windows and Motif (left) and Macintosh (right).

3.1.1.2.1. Border

The **border** is a frame that surrounds the window and contains all the window elements. In Motif and Windows, all borders and corners can be grabbed by the user with the cursor and used to resize the window. In Apple only the resize box (an optional element) in the bottom right corner can be used to resize the window.

√ Every window must have a border.

3.1.1.2.2. Client area

The **client area** is where the controls for the application are placed. Sections 4 and 5 provide guidelines and examples on designing the client area of an application.

√ Every window must have a client area.

3.1.1.2.3. Control menu

The **control menu** is a menu of all possible actions for the window. See Section 3.4.2.1 on menus below. (The control menu is called the "window menu" in Motif.) Apple does not have this object.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Locate the control menu button in the leftmost region of the title bar.

+ In Motif and Windows, it is *strongly recommended that every window* have a control menu.

3.1.1.2.4. Title

Titles are short verbal descriptions of the contents of the window.

√ Every window with a title bar must have a title -- center the title in the title bar at the top of the window. For command completion dialog boxes, begin the title in the upper left corner of the client area. Usually the title of the command completion dialog box is the same as the command that initiated the posting of the dialog box.

√ For primary windows, the title must be *application name - file name*; for example, "BestWrite - Pulleys." For secondary windows, the title must begin with the object under focus and the action; for example, "Pulleys - Find" indicates the user is doing a find operation in the document "Pulleys." "Printer - Set Up" is the title of the secondary window for the action of setting up the printer (an object).

√ If the user opens more than one instance of the object (e.g., opens the same document twice), title each instance with a colon and an instance number attached. For example, when the second instance is opened, ":1" is appended to the title of the first instance, e.g., "BestDB - Guidelines:1", and ":2" is included in the second instance of the title, e.g., "BestDB - Guidelines:2".

√ All significant words in the title, except for user-defined words, should have their initial letter capitalized.

√ In Apple, floating palettes do not have a title.

+ It is acceptable to include the machine and/or path in the title bar.

Δ If a database application opens multiple views *of the same object* (i.e., same database) titling is trickier. For example, if the user displays a list of just the names and addresses from the fundraising database, then wants to create another list from the same database of names, phone numbers and amounts, these are different views of the same database. One solution is to prompt the user for a name (naming should not be required, however) each time a separate view of the object is requested, and post that name in the title as follows: *application - file*

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

name - view name. Continuing the example, if the user entered the name MAILING LIST then the title of that window should be BESTDB - FUNDRAISING - MAILING LIST. This solution should work in most cases.

More reading

Apple HIG, p. 42-43.
OSF/MSG, § 5.3.
IBM SAA CUA AIDR, p. 272-276.

3.1.1.3. Optional elements

“Optional” means that you may not need one or more of the elements discussed below. However, *if* you do use one or more of the elements then they must behave in the standard way described below.

The optional interface elements for Apple, Windows, and Motif are discussed separately below.

3.1.1.3.1. Apple

Figure 3-2 shows graphically the optional elements of an Apple window.

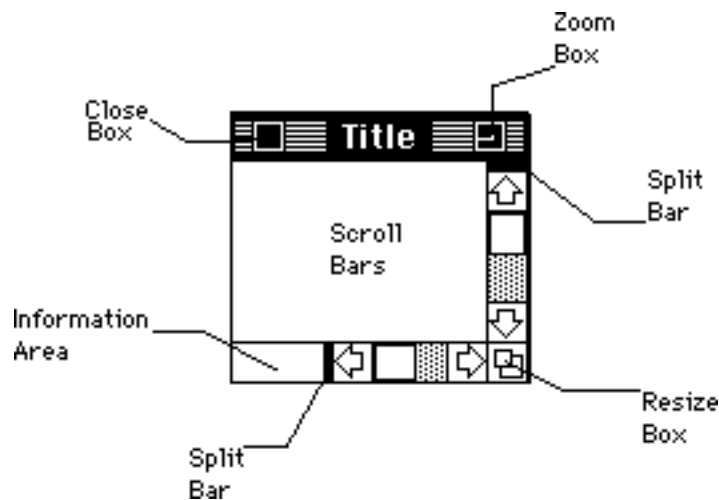


Figure 3-2. of an Apple window.

Close box

The **close box** is a square box in leftmost region of the title bar. If the user clicks in the close box the window goes away.

√ If the contents of the window have been modified and not saved and the close box is clicked, then prompt the user with the standard save

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

changes dialog box (see Section 3.4.3.3). *Exception:* In some applications, where the application is monitoring a process, changes appear in the window, but there is nothing to save (e.g., Print Monitor).

Zoom box

The **zoom box** is in the rightmost region of the title bar. If the user clicks on the zoom box the window reverts to its previously selected size. If clicked again it returns to its default size. On a new window, the "previously selected size" is determined by the application: it could be equal to the default size or it could be larger or smaller.

Split bar

Split bars are initially placed at the top of the vertical scroll bars and the left of horizontal scroll bars. The user can click and drag the split bar to divide the window into panes anywhere along the scroll bar.

√ After the split, scroll bars appear on either side of the split bar so that each pane can be manipulated independently.

Scroll bars

Scroll bars are located on the far right and bottom of the window. They are used when the size of the client area is not large enough to present all of the application document. Scroll bars are discussed in more detail in Section 3.1.1.5.1.

Size box

The **size box** is located in the lower right corner of a window. It allows the user to click on it, and, by dragging the corner, resize the window.

√ Only windows with scroll bars have size boxes.

Apple allows a close box, a zoom box, a size box, and the ability to drag a window. Table 3-1 shows the different window types defined by combinations of these elements.

Information area

The **information area** provides information on settings being used by the application or on objects being manipulated. Text placed in the information area should be non-essential, but nevertheless helpful to users. See Section 5.3.3.2 for more information on information areas. Many popular shrink-wrap Macintosh software applications, such as Microsoft Word 4.0, use information areas.

+ For almost all applications, put an information area to the left of the horizontal scroll bar or on the very bottom of the window below the scroll bar.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Apple Window Types	Close Box	Zoom Box	Size Box	Draggable	Used for
Standard	Optional	Usually	Usually	Yes	Secondary windows and sometimes for applications
Scrolling	Usually	Usually	Usually	Yes	Primary windows in applications
Round rectangle	Yes	No	No	Yes	Desk accessories
Dialog	No	No	No	Yes	Dialog boxes
Floating	Yes	No	No	Yes	Palettes

Table 3-1. Apple window types defined by optional elements.

3.1.1.3.2. Motif

Figure 3-3 shows the optional elements of a Motif window.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

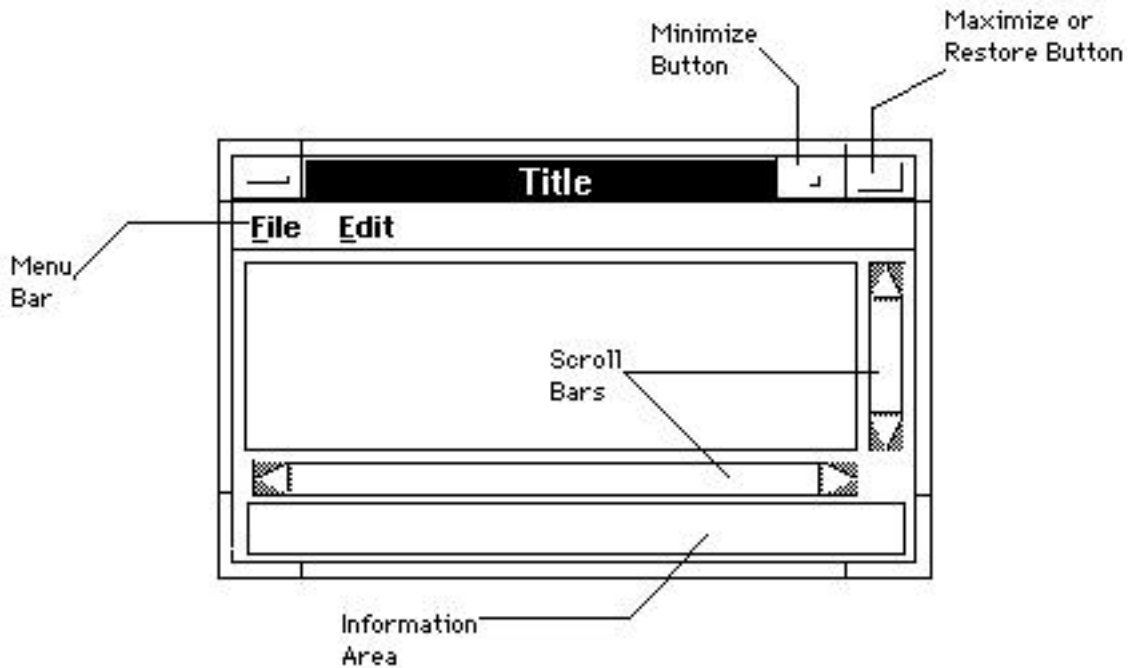


Figure 3-3. of a Motif window.

Maximize button

The **maximize button** is located in the title bar in the right most region of a primary window; secondary windows do not have maximize buttons.

√ When the maximize button is picked, the window is enlarged to its largest size. The maximize button must be enabled when the window can be resized but is not currently maximized. When the window is maximized, change the maximize button icon to a restore button icon.

+ To be consistent with Windows, it is recommended that the used for the maximize button be the up arrow (not the large square -- see OSF/Motif § 5.3.3).

Minimize button

The **minimize button** is located in the rightmost area of the title bar of the primary window, just to the left of the maximize/restore button; secondary windows do not have minimize buttons.

√ When the minimize button is picked, remove the window and all secondary windows from the screen, and place an application-defined icon on the desktop.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ To be consistent with Windows, it is recommended that the used in the minimize button is the down arrow (not the small square -- see OSF/Motif § 5.3.3).

+ Sometimes minimize is referred to as “iconize” or “iconify”; the term “minimize” is strongly recommended.

Restore button

The **restore button** is located in the rightmost area of the title bar in the primary window.

√ When the restore button is picked, return the window to the size and location it had before it was previously minimized or maximized. Also, when restoring from a minimized state, restore any secondary windows associated with the minimized window. The restore button icon replaces the maximize icon when the window is maximized and the window can be resized. Change the restore button icon to the maximize button icon when the window is not maximized and the window can be resized.

+ To be consistent with Windows, it is recommended that the restore button be a vertical double-pointed arrow (see OSF/Motif § 5.3.3).

Menu bar

The **menu bar** is the area just below the title bar that contains one-word menu titles for associated pull-down menus.

√ Use menu bars when the window uses the standard menus File, Edit or Window menus (see Section 3.4.2). If the menu bar is not used, all actions must be provided on buttons in the client area.

Sash

The **sash** is used to adjust the border between two groups of components in a paned window. It contains a handle on the separator that allows users to adjust its location within the window.

Scroll bars

Scroll bars are located on the right and bottom of the client area. They are used when the size of the client area is not large enough to accommodate all of the objects in the client area. Scroll bars are discussed in Section 3.1.1.5.1.

Information area

For almost all applications, put an **information area** (Motif calls it a “message area”) to the left of the horizontal scroll bar or on the very bottom of the window. The information area provides user guidance and/or information on settings being used by the application or on objects being manipulated. See Section 5.3.3.2 on the use of information areas.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Command area

The **command area** is for entering commands. The command area should be located at the bottom of the client area below the scroll bar, and above the information area if that is located at the bottom of the display.

△ The command area should be used sparingly, but may be useful for some applications where users need to enter commands. For example, if the users are familiar with the UNIX[®] operating system they may find it more efficient to perform certain actions on the command line (e.g., piping and filtering) than with the graphical user interface.

3.1.1.3.3. Windows/CUA

Figure 3-4 shows the optional elements of a Windows window.

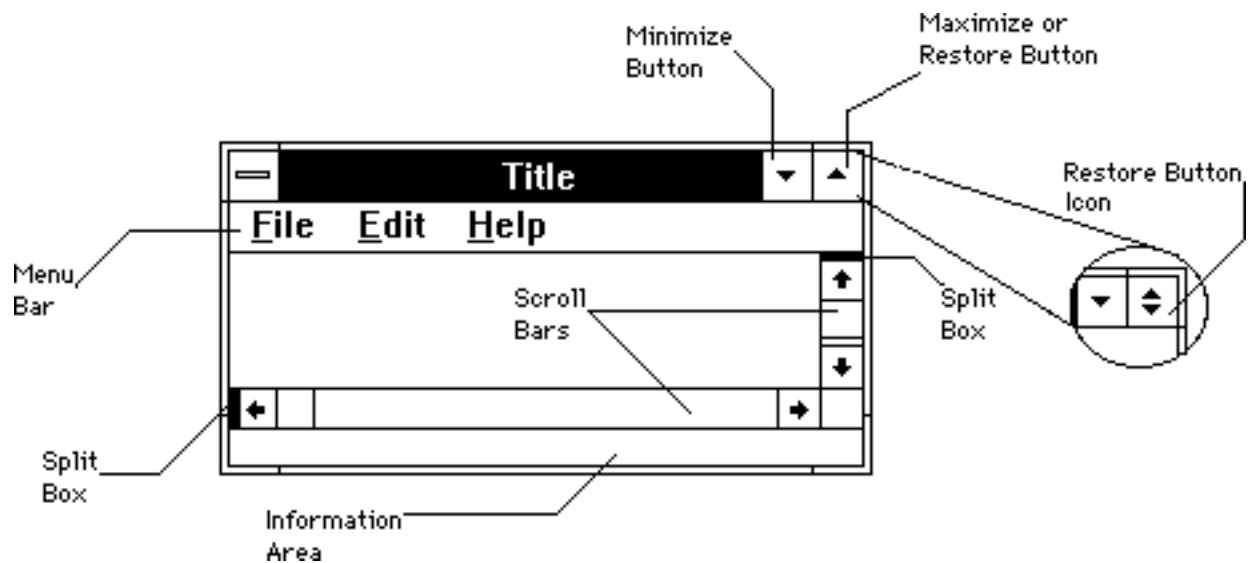


Figure 3-4. of a Windows' window.

Maximize button

The **maximize button** is located in the rightmost area of the title bar of primary windows. Fixed size windows do not have maximize buttons.

√ When the maximize button is picked, enlarge the window to its largest size or to the size of the display screen, whichever is *smaller*. The maximize button must be enabled when the window can be resized, but is not currently maximized. When the window is maximized, change the maximize button to a restore button. Secondary windows do not have maximize buttons.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Minimize button

The **minimize button** is located in the rightmost area of the title bar of primary windows, just to the left of the maximize/restore button.

√ When the minimize button is picked, remove the window and all secondary windows from the screen, and replace it with an application-defined icon on the screen. Secondary windows do not have minimize buttons.

+ Sometimes minimize is referred to as "" or "iconify"; the term "minimize" is strongly recommended.

Restore button

The restore button is located in the right most area of the title bar. The restore button appears when the window is maximized and the window can be resized.

√ When the restore button is picked, return the window to the size and location it had before it was maximized. Also, when restoring from a minimized state, restore any secondary windows associated with the minimized window. Change the restore button to the maximize button when the window is not maximized and the window can be resized. Secondary windows do not have restore buttons.

Menu bar

The **menu bar** is the area just below the title bar that contains one-word menu headings for associated pull-down menus.

√ Use menu bars when the window uses the standard menus: File, Edit and Window (see Section 3.4.2). If the menu bar is not used, all actions must be provided through window controls or on push buttons in the window.

Split box

A **split box** appears at the top of the vertical scroll bars and the left of the horizontal scroll bars. The user can click on the split box and then drag the split bar to divide the window into panes anywhere along the scroll bar.

√ After the split, scroll bars appear on either side of the split bar so that each pane can be manipulated independently. Windows without scroll bars cannot have a split box.

Scroll bars

Scroll bars are located to the right and bottom of the client area. They are used when the size of the client area is not large enough to accommodate all of the objects in the client area. Scroll bars are discussed in more detail in Section 3.1.1.5.1.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Information area For almost all applications, put an **information area** (Windows calls it a “message bar or a status bar”; see § 4.2.6 and 4.2.7 or the Windows ADG and Section 5.3.3.2 of this document) at the bottom of the display, below the scroll bars or to the left of the horizontal scroll bar.

+ The information area provides user guidance and/or information on settings used by the application or on objects being manipulated.

More reading

Apple HIG, p. 43-54.
OSF/MSG, § 4.2.1, 5.3.
Windows ADG § 4.1, 4.2
IBM SAA CUA AIDR, p. 119-120, 136, 146, 209-210, 216-217, 249, 272-276.

3.1.1.4. Properties

3.1.1.4.1. Active/Inactive

A window is made **active** by clicking anywhere inside the window frame or picking it from the window menu.⁵ The active is the window with which the user is working at any given time; it has the input focus. When activated, the title bar is operative, the window and its contents become visible, and all window elements appear. Anything deselected when the window was made inactive is selected again. Figure 3-5 shows an example of an active and inactive windows display.

When a window is made active, all other windows are deemed **inactive**, although there may still be operations (i.e., background processing) occurring in that . When the window becomes inactive, selections are deselected, the title bar is inoperative, and the other window elements disappear or change appearance (e.g., supplemental windows -- floating palettes). See the relevant style guides for exact details of activating and de-activating windows.

⁵ Described here is the **explicit** focus model. There is another model called the **implicit** focus model. In the implicit focus model a window is activated simply by positioning the cursor in the window. Windows and Apple use the explicit focus model. In Motif environments, the focus is an environmental variable out of control of the application. It is recommended, however, that the explicit focus model be used in most cases, especially for inexperienced users.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES



Figure 3-5. An example of active and inactive windows.

More reading

Apple HIG, p. 45.
OSF/MSG, § 2.1.
Windows ADG, § 4.3.5.
IBM SAA CUA AIDR, p. 32-33, 115.

3.1.1.5.

3.1.1.5.1. Within a window

Scroll bars enable users to view any part of the client area that does not fit in the current display. If the client area has no information to scroll to then make the scroll bars inactive.

The scroll bar is a rectangle with a scroll arrow on either end. A scroll box sits inside the scroll bar and shows the relative location, in the entire document, of the portion of the document currently visible. Vertical scroll bars scroll the window up and down; horizontal scroll bars scroll the window left and right.

The user manipulates the scroll bar (and thus the client area) by either clicking and dragging the scroll box, clicking on the scroll arrow, or clicking in the rectangle above or below the scroll box. See the relevant style guides for exact navigational principles.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Make scroll box sizes represent the relative amount of the document displayed in the window. If the document is short, the displayed amount of the document is large and thus so is the scroll box. If the document is long, then the displayed amount and scroll box are small.

Motif has proportional scroll boxes built into the scroll bar widget. Proportional scroll boxes do not violate the Apple HIG (see page 51), but they are not commonly found in most applications. Some shrink-wrap applications (e.g., Click Change) for the Macintosh will make scroll bars proportional. Proportional scroll bars are required in CUA applications, and should be used for Windows applications.

Note: The only drawback to proportional scroll boxes is that for an extremely long document the scroll box will be very small. Therefore, the minimum size of the scroll box should be the default scroll box size. If the entire document fits within the window the scroll bars should be inactive.

3.1.1.5.2. *Moving windows on the display*

Users move windows by clicking on the and dragging a frame of the window to the new location. When the user releases the mouse button, the window is redrawn in the new location.

√ A window must never be able to be moved off the display such that it cannot be seen, although it may be completely hidden by one or more other windows.

3.1.1.5.3. *Among windows*

Users can have several applications, primary windows, and secondary windows available at any given time. Thus, the user must be able to quickly and easily navigate among the windows. The most straightforward way that users will navigate among windows is to click on an inactive window with the mouse pointer and make that window active.

Another way to activate a window is to select its name from the Window menu in an application (see Section 3.4.2.5).

Both Windows and Motif provide a means of navigating among windows by using (e.g., in Windows, ALT+TAB or ALT+ESC moves the active window to the bottom of the window stack and moves the "next" window to the front), by using the (Motif) or Switch To (Windows) in the control menu. Apple allows users to pick from current applications (in System 7.0) from the Finder icon in the upper right corner of the desktop.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 42-54.
OSF/MSG, § 2.1, 4.2.1.3, 5.4.
Windows ADG, § 4.3.1, 4.3.5, 4.3.6.,
IBM SAA CUA AIDR, p. 216-217.

3.1.2. Dialog boxes

Dialog boxes are a type of secondary window. Section 3.1.1 discussed the required elements for windows. However, because dialog boxes are specialized in many respects they deserve more detailed treatment.

There are two general uses for dialog boxes. First, dialog boxes are used to get more information to complete a command, termed "Command completion dialog boxes". Second, dialog boxes are used to present a message to the user and get the user's reply. These are categorized as information, warning, and error dialog boxes.

√ Dialog boxes are not sizable.

3.1.2.1. Types

3.1.2.1.1. Command completion

Command completion dialog boxes are used to gather more information from the user before a command (that the user issued) can be completed. Command completion dialog boxes can be modal or modeless (see Sections 3.1.1.1.2.1 and 3.1.1.1.2.2).

Motif has standard widgets that apply to some command completion dialog boxes, such as the fileselectiondialog. Those that can be standardized across platforms are described later in Sections 3.1.2 and 3.4.3. For those that are not standardized, Section 4.4 discusses in detail the guidelines for selecting objects and the layout of the client area of the command completion dialog box.

√ Labels on command items (e.g., buttons and menu items) that generate dialog boxes must be followed by an ellipsis (...).

3.1.2.1.2. Information boxes

An **information box** is used to provide status information, make helpful suggestions, give concrete advice, and to get responses to simple

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

questions. (*Note:* This definition is slightly *less* restrictive than in the Windows ADG.)

√ Information boxes are modal. The user must pick one of the available buttons in order to continue.

+ Most information boxes have three main elements: an icon, a message, and a set of buttons. They are arranged with the icon on the left, the message to the right of the icon, and the buttons below the message.

+ Information boxes that provide critical information should be accompanied by a beep.

Note: Most times you will want to provide a default button with information boxes, but be aware that providing a default button is not always correct or desirable, such as when it is critical that the user reads the message. An impatient user or a user who accidentally presses RETURN could bypass a crucial message and not have that important information available. If a default button is used, it should be the response that will cause the least destructive action to take place and should be selected if the user presses RETURN.

Figure 3-6 shows an example of an information dialog box.

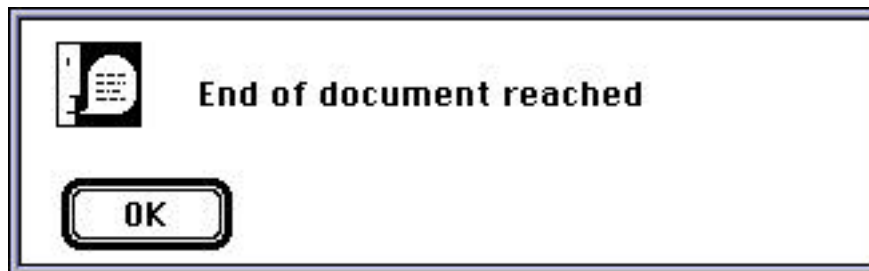


Figure 3-6. Information dialog box.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.2.1.3. Warning boxes

Warning boxes call the user's attention to an operation that may have undesirable consequences if it continues or if the system is not sure that the process terminated normally (e.g., "Print job may not have completed properly; check output and re-print if necessary"). Usually warning boxes will allow the user to continue the action, re-try it, or cancel it.

√ Every warning box must have three main elements: an icon, a message, and a set of buttons. They are arranged with the icon on the left, the message to the right of the icon, and the buttons below the message. If a default button is desired, it should be the response that will cause the least destructive action to take place and should be selected if the user presses RETURN.

√ Warning boxes are modal. The user must pick one of the available buttons in order to continue.

√ Warning boxes must be accompanied by a beep from the application.

+ Warning boxes should have an option that allows the user to withdraw the action that caused the warning condition, such as `Cancel`, see Figure 3-7.

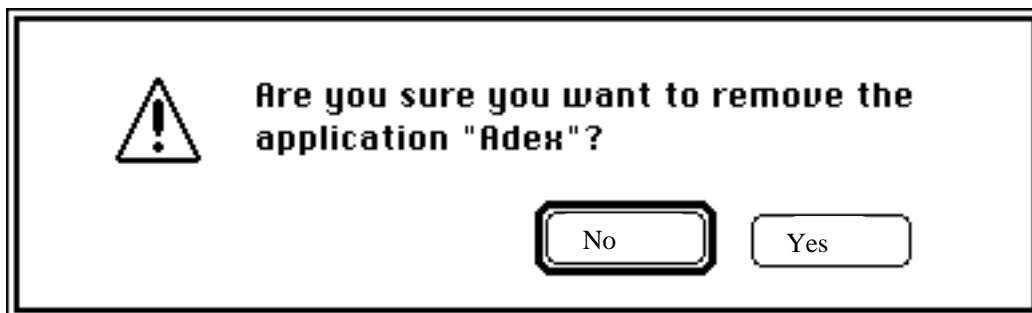


Figure 3-7.

+ Warning boxes should have an option that allows the user to continue the action that caused the warning condition.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.2.1.4. Error boxes

Error boxes (Windows calls them "Critical" message dialogs) inform the user of a serious system or application problem (see Figure 3-8). Error boxes often require the user to choose from alternative courses of action.

√ Every error box has three elements: an icon, a message, and a set of buttons. They are arranged with the icon on the left, the message to the right of the icon, and the buttons below the message. If a default button is desired, it should be the response that will cause the least destructive action to take place and should be selected if the user presses RETURN.

√ Error boxes are modal; the user must respond to the message by picking one of the available buttons in order to continue.

√ Error boxes must be accompanied by a beep from the application.

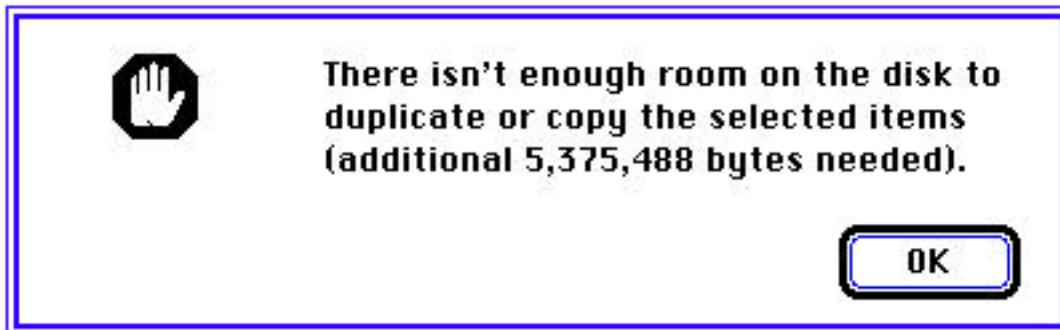


Figure 3-8. An example of an error dialog box.

More reading

Apple HIG, p. 58-62.
OSF/MSG, § 4.1.1, 4.2.1.5, 4.2.4.
Windows ADG, § 7.1.4
IBM SAA CUA AIDR, p. 31, 221-222.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.2.2. Standard dialog boxes

Dialog boxes help simplify commands and provide more information to the user about actions. They are transitory. Dialog boxes come in many forms with a variety of layouts, but most are one of the basic types found in Table 3-2. The remainder of this section presents standard dialog boxes. Section 3.4.3 must be consulted for specific Ameritech standard dialog boxes.

General Type	If your purpose is to...	Then use...
Command completion	allow the user to make simple and predictable changes for command completion	OK/Cancel dialog box
	allow the user to make changes to properties of an object where immediate feedback is desirable in command completion	Property dialog box
Information	convey to the user important system or application information; present information for the user to confirm	Message dialog box
	provide the user with a means of monitoring dynamic data	Close dialog box
	get user response to a question	Question dialog box
	inform the user that the application is working	Working dialog box
Warning	alert the user to a condition that requires a decision or some input before it can proceed, e.g., overwriting a current file	Warning dialog box
Error	convey information about an error	Error dialog box

Table 3-2. Standard dialog box types.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

The balance of this section describes the contents and form of many common dialog boxes. Section 4.4 describes how to lay out dialog boxes.

3.1.2.2.1. Command completion dialog boxes

OK/Cancel Dialogs

OK/Cancel dialog boxes are used for confirming changes that the user requests, such as in Figure 3-9. Changes are effected when the user presses **OK**. All settings or further processing is canceled if the user presses **Cancel**.

√ The dialog box goes away when the user selects either button.

+ Use an OK/Cancel dialog box for predictable changes; that is, when the user can predict how the change will look or how the action will proceed.

Δ An OK/Cancel dialog box can also have a **Help** button.

+ The following button sequences are recommended for OK/Cancel dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

OK
OK Cancel
OK Cancel Help

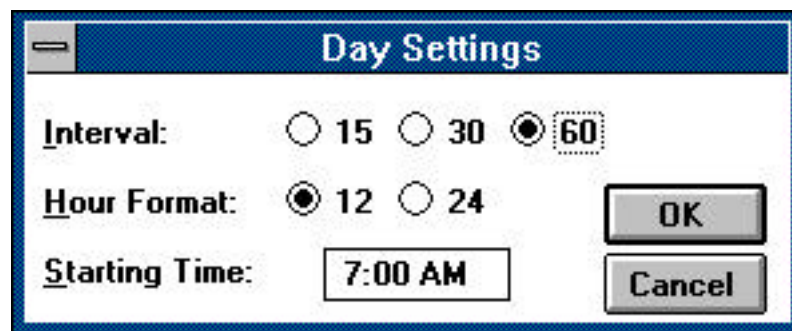


Figure 3-9. An example of an OK/Cancel dialog box.

Property Dialogs

The main difference between the Property dialog box and the OK/Cancel dialog box is that **Property dialogs** show the effects of applying the commands immediately without removing the dialog box; OK/Cancel dialog boxes are removed when the action is taken. By

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

pressing `Apply` the user can effect a change immediately. Figure 3-10 shows an example.

+ An `Undo` button is strongly recommended when there is no way to easily reverse an action by reselecting the previous choice. When using `Undo`, allow the user to `Redo` an undone action. For example, if the user makes a change, then undoes that change, the `Undo` button should change wording to `Redo`. So by repeated pressings of `Undo/Redo` the user can rapidly flip between two possibilities until another action is taken.

+ The title of a Property dialog should be the name of the command that launched the dialog box. Figure 3-10 is titled "Paragraph" which comes from selecting the `Paragraph...` menu item.

Δ A good practice to follow is to label one of the buttons on a dialog box with the same label or a similar label as the button or menu item that launched the dialog box. For example, if the user selects `Open...` from the File menu, there should be a button labeled `Open` on the dialog box instead of a button like `OK` or `Apply`.

Δ Use the Property dialog box, as opposed to the `OK/Cancel` box, when the user needs to see immediate feedback of dialog box actions. The `Done` button (instead of `OK`) makes the dialog box go away. This kind of dialog box may have a `Help` button. The following button sequences are recommended for property dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Apply Done
Apply Done Help
Apply Undo/Redo Done
Apply Undo/Redo Done Help

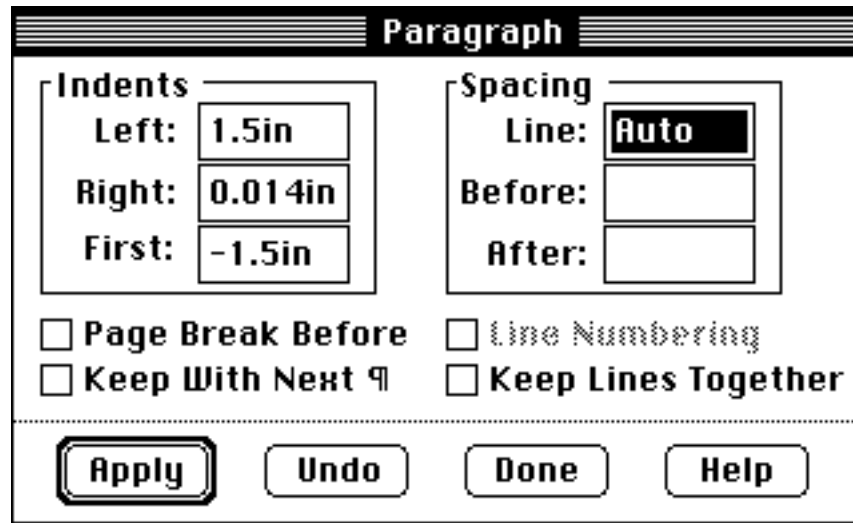


Figure 3-10. Example of Property dialog box.

3.1.2.2.2. Information dialog boxes

Message Dialogs

Message dialog boxes are used for passing information to the user. Pressing an affirmative button is the user's way of signaling that she or he has read the message. System or application information the user needs to know is passed in message dialog boxes. An example is presented in Figure 3-11. The following button sequences are recommended for message dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

OK		
OK	Help	
OK	Cancel	
OK	Cancel	Help
Retry	Cancel	
Retry	Cancel	Help
Action	Don't	
	Action	
Action	Don't	Help
	Action	

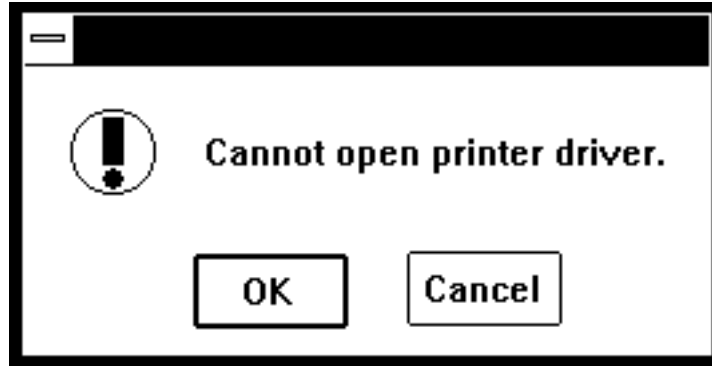


Figure 3-11. Example of a message dialog box.

Close Dialogs

Close dialog boxes are used for monitoring dynamic data or for displaying captured images.

Close dialog boxes are modeless and do not assume that the user has processed any information in the dialog box. A clock is an example, such as in Figure 3-12. "Close" and "move" are the only actions available for this dialog box. In Macintosh, use a dialog box that only has a close box in the title bar; in Windows and Motif, use the `Close` and `Move` items on the control menu.

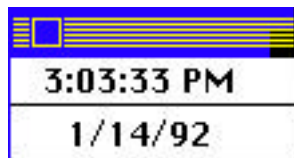


Figure 3-12. Example of a Close dialog box.

Question Dialogs

Question dialogs require the user to respond to a question involving non-destructive actions. Figure 3-13 shows an example. Question dialogs

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

are modal; the user must answer before continuing. The following button sequences are recommended for Question dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

Yes	No		
Yes	No	Help	
Yes	No	Cancel	
Yes	No	Cancel	Help

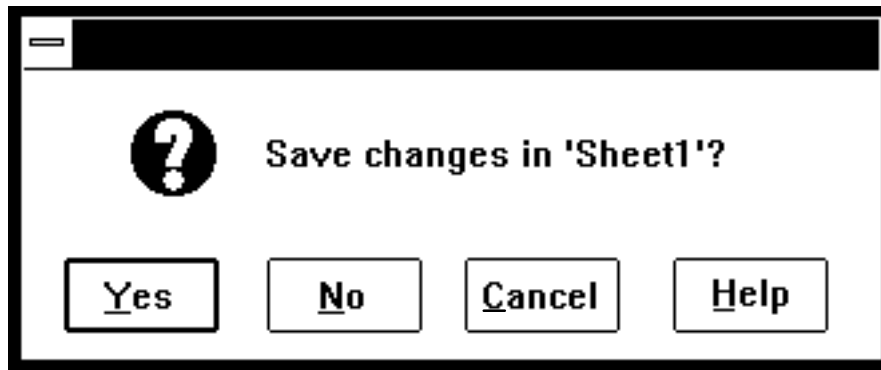


Figure 3-13. Example of a question dialog box.

Working Dialogs

Working dialogs are progress indicators that allow the user to cancel ongoing processing. Figure 3-14 shows an example. Working dialogs, where possible, are modeless. When the user presses `Cancel` the process quits and may require the application to inform the user of the status of the process. For example, if the user cancels a copy operation during the copy of the 21st of 40 files then when the user presses `Cancel` the first 20 should be in the new location and the other 20 (including the one in progress) should be in the old location. The following button sequences are recommended for Working dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Cancel
Cancel Help

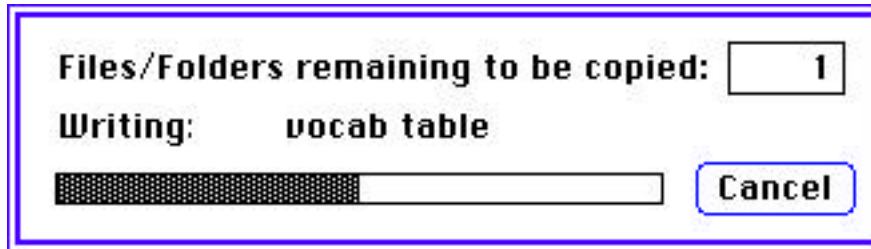


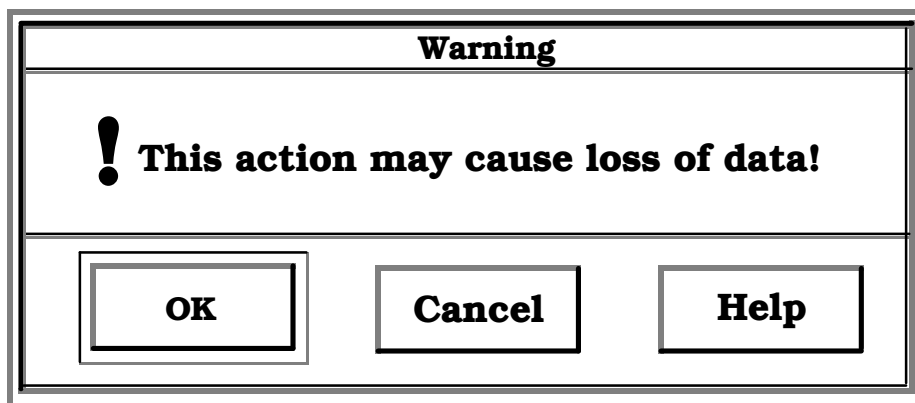
Figure 3-14. Example of a working dialog Box.

3.1.2.2.3. Warning dialog boxes

Warning Dialogs

Warning dialogs tell the user of a potential destructive or non-reversible action. The user can choose to continue the process if she or he desires. Figure 3-15 shows an example. Warning dialogs should be modal, and contain the standard system warning icon. The following button sequences are recommended for warning dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

Yes	No	
Yes	No	Help
Continue	Cancel	
Continue	Cancel	Help
OK	Cancel	
OK	Cancel	Help



AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Figure 3-15. Example of a Warning dialog box in Motif.

3.1.2.2.4. Error dialog boxes

Error Dialogs

Error dialogs tell the user of a serious system or application problem. Figure 3-16 shows an example. They should be modal, and contain the standard system error icon. The following button sequences are recommended for error dialog boxes. An explanation of each of these standard command button actions is found in Table 3-3 in Section 3.1.2.2.6.

OK
OK Help
Continue
Continue Help

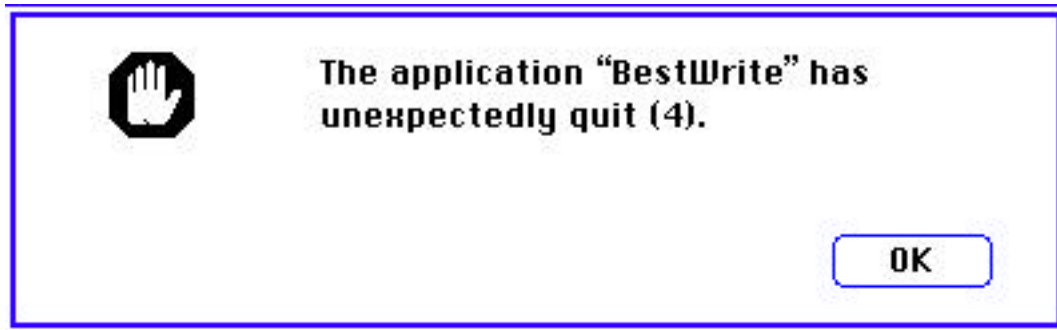


Figure 3-16. Example of an Error dialog box in Apple.

3.1.2.2.5. General recommendations for

- √ A dialog box must not have more than one default button.
- √ Make the default button the least destructive option. See Section 4.2.3.3 for more information on default buttons.
- √ The default button must be the leftmost or topmost button in the set of buttons.
- √ Always make dialog boxes completely visible on the primary display. If covering underlying information is a problem, use movable dialog boxes.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ When a dialog box is displayed, show the current settings of all the controls on the dialog box. For example, if the user can make the same settings in two places in the interface (e.g., through a dialog box or through keyboard commands) then each location should always reflect the current settings irrespective of where or how that setting was made.

+ A dialog box should be opened near the control or condition that caused the dialog to be displayed. Thus, first try to position the dialog box to the right of the control if there is not enough space, then place the dialog box beneath the object. If you cannot fit the entire dialog box on the display, center the dialog box on the display even if it obscures the control that caused the dialog to be displayed.

+ Place application specific buttons first, (if applicable) then *OK*, *Apply*, *Reset*, *Cancel*, and *Help*.

3.1.2.2.6. *Standard dialog command buttons*

Table 3-3 shows common dialog box buttons and their meanings.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Button	Meaning
Yes	Affirmative answer to a question. When selected, it dismisses the dialog box. Use <i>Yes</i> only if it is a precise and correct answer to the question.
No	Negative answer to a question. When selected, it dismisses the dialog box. Use <i>No</i> only if it is a precise and correct answer to the question.
OK	Causes changes made in the dialog box to be applied to the application or indicates that the user has read and understood a message. When selected, it dismisses the window. <i>Done</i> may also be used, if it makes more sense than <i>OK</i> . Generally, <i>OK</i> should not be used in reply to a question, use <i>Yes</i> and <i>No</i> .
Done	Same as for <i>OK</i> .
Apply	Applies the changes made in the dialog box, but, unlike <i>OK</i> , does not dismiss the window.
Action	Performs action implied by its text. If the <i>Action</i> button is the only button on the dialog box, perform the action and close the dialog box. For example, "Save" and "Repage" are actions. <i>Don't Action</i> is used to indicate the opposite case of <i>Action</i> .
Undo	Undoes the changes made by <i>Apply</i> , <i>OK</i> , <i>Done</i> , or <i>Yes</i> ; does not dismiss dialog box. The button is changed to <i>Redo</i> when it is selected.
Retry	Causes the action in progress (or just completed) to be attempted again. <i>Retry</i> could be used in error dialogs, for instance when the user needs to input a diskette.
Cancel	Closes dialog box without performing user changes. If used with <i>Apply</i> , it does not affect any changes already applied.
Options >>	Many applications will find that, especially in command completion dialogs, that there are some features that are basic and others that are more advanced. The dialog should initially come up in the basic state with an <i>Options >></i> button. When the user presses <i>Options >></i> the dialog box increases in size and the advanced features appear. The <i>Options >></i> button should change to <i>Options <<</i> to indicate that pressing it will return to the basic state.
Help	Displays context sensitive help for the entire dialog box and its contents.

Table 3-3. Common command buttons used in dialog boxes.

More reading

Apple HIG, p. 60-62
OSF/MSG, § 4.2.1.5
Windows ADG, § 7.1, 7.2, 7.3
Kobara, p. 223-228
IBM SAA CUA AIDR, p. 29-30, 121, 270-271
IBM SAA CUA AIDG, p. 77-81

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.3. Menus

Menus are collections of commands, attribute selections, separators, and other elements that are selectable. Most primary windows will have a menu bar that organizes a number of pull-down menus. The items that appear in the menu bar do not have any actions associated with them aside from displaying the pull-down menu. Pull-down menus, in turn, may have secondary menus, called cascaded menus. Some menus can be “torn-off” from a pull-down menu on the menu bar and reside as a window on the display; these menus are called “tear-off menus.”

3.1.3.1. Types

There are a wide variety of menu types. However, for purposes of designating standards only two of those types are of concern here: pull-down menus and cascaded menus. Other menu types are discussed in terms of their appropriate usage in the client area of the window (see Sections 4.1 and 4.2).

3.1.3.1.1. Pull-down menu

Pull-down menus show a set of commands and attribute choices that are related to the menu bar item. Figure 3-17 shows an example of a pull-down menu.

3.1.3.1.2. Cascaded menu

Cascaded menus (sometimes called “walking menus”) contain sets of choices that are related to the menu item that generated their presentation. Figure 3-17 shows an example of a cascaded menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

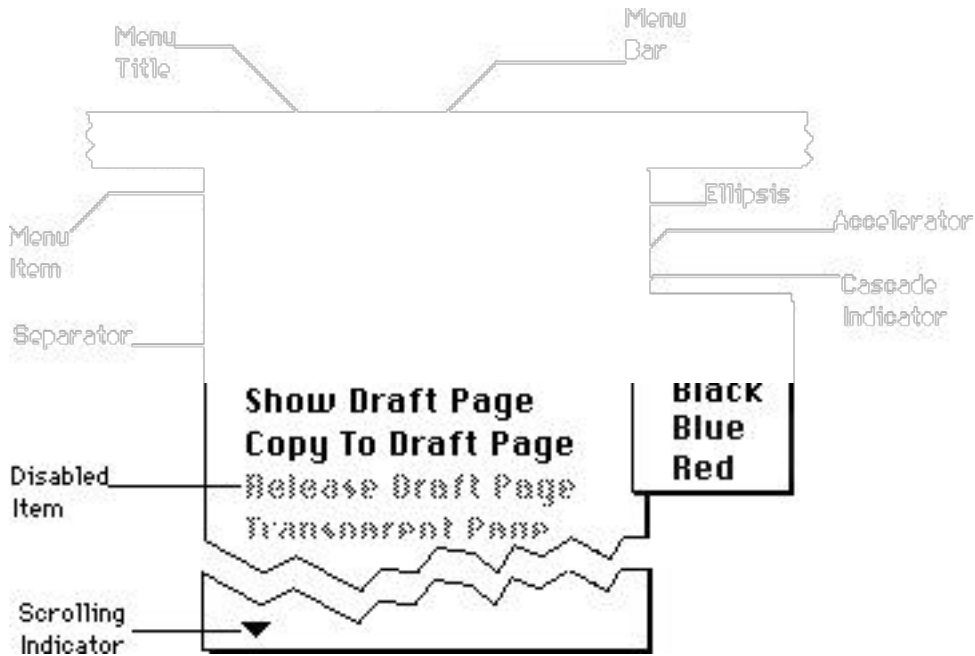
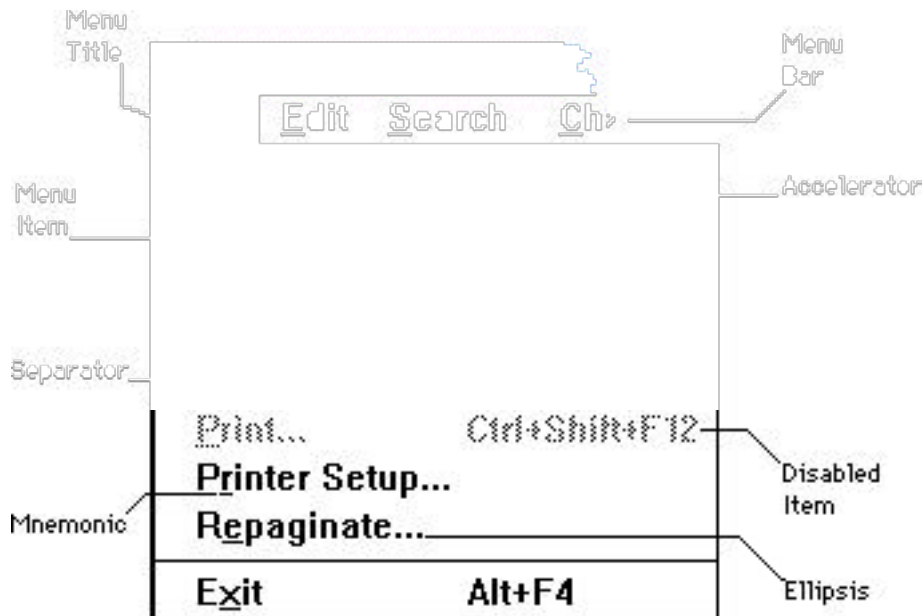


Figure 3-17a. Example of an Apple menu.



LEGEND: √ Required +Recommended Δ Good Practice

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Figure 3-17b. Example of a Windows menu.

More reading

Apple HIG, p. 65-73, 87-88.
OSF/MSG, § 4.2.1.4, 4.2.3.
Windows ADG, § 5.1, 5.2.
IBM SAA CUA AIDR, p. 36, 138-139, 193.

3.1.3.2. Required elements

3.1.3.2.1. Menu bar

The **menu bar** is a horizontally arranged collection of menu titles located at the top of the window or display.

√ Locate the menu bar at the top of the application just below the title area of the window in Windows and Motif, and at the top of the display in Apple.

√ All primary windows must have a menu bar with the required menus.

√ The menu bar must extend the full width of the primary window.

√ Do not use multiple row menu bars when the application is in its standard state.

3.1.3.2.2. Menu titles

Menu titles are short descriptive single-words that reflect the kinds of commands present in its pull-down menu.

√ If all the items in the menu are disabled (i.e., they cannot be selected; graphically they are), then disable the menu title (see Section 3.1.3.4.1). However, users must still be able to pull down a menu and see-the disabled menu items even if the menu is disabled.

+ It is *strongly* suggested that menu titles be single words.

3.1.3.2.3. Menu items

Menu items are commands and attributes that the user has available to control the application.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Unavailable items must not disappear; they are disabled, i.e., grayed out (see Section 3.1.3.4.1).

√ Do not add or remove items from a menu unless the user takes explicit action to add or remove the item through the application.
Exceptions: The label for a specific menu item might toggle between two different states (see Section 4.1.1.1.5). Also, Microsoft allows for a “most recently used (MRU) list of files” (Windows ADG, §5.4.2.2). The MRU provides quick access to recently used files, without requiring the user to work through a dialog box. Using an MRU is encouraged.

√ The menu itself must be wide enough to accommodate the longest menu item and its accelerator, ellipsis, or cascade indicator (see below).

+ A menu item should not have the same wording as its menu title.

3.1.3.2.4. Mnemonic

A **mnemonic** is a single underlined character in a menu title or menu item label. Mnemonics provide quick access to Windows and Motif menus and menu items from the keyboard. Apple does not support mnemonics. See Section 3.4.2 for mnemonics for standard menu items.

√ Every menu title and menu item in Motif and Windows must have an underlined mnemonic character.

√ Do not number menu items, except when the number of items on the menu can vary as with a Window menu (see Figure 3-18, and Section 3.4.2.5). The number then serves as the mnemonic.

+ The preferred mnemonic is the first character of the menu item’s label. If that letter conflicts with another menu item’s mnemonic, the next character in the label should be used.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

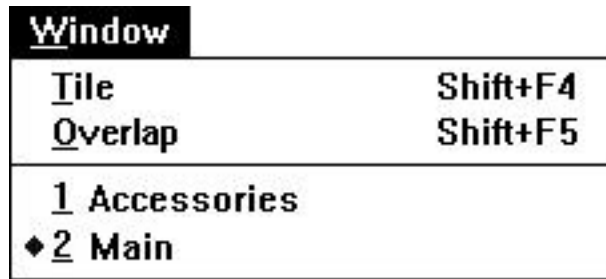


Figure 3-18. Example of a Windows Window menu. As each new group is added to the list, it is given a number that becomes the menu item's mnemonic.

3.1.3.2.5. Ellipsis

An **ellipsis** (...) indicates to the user that a command completion dialog box will follow selection of the menu item.

√ Every menu item that is followed by a command completion dialog box must have ellipsis following the item.

3.1.3.2.6. Cascade indicator

A **cascade indicator** tells the user that a sub-menu is displayed when the item is selected. It is shown as a right pointing arrow head (→) at the far right portion of the menu item. If the user selects the menu item, the cascaded menu appears next to the menu item. See the relevant style guides for exact navigational principles.

√ Every cascaded menu must be signaled with a cascade indicator.

3.1.3.2.7. Separators

Separators are subtle lines on the menu itself used to visually divide groups of related menu items .

+ Use separators to distinguish groups of related choices.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.3.2.8. Scrolling Indicator

Scrolling indicators are up (▲) or down (▼) pointing arrow heads cueing the user that there are more menu items to be displayed if the user selects the scrolling indicator. If the user selects the scrolling indicator, the menu scrolls in the direction of the arrow. See the relevant style guides for exact navigational principles.

√ If a menu has more items than can be displayed, place a scrolling indicator at the top and/or the bottom of the menu. Place the scrolling indicator at the top if more menu items are available before the first displayed item place a scrolling indicator at the bottom of the menu if more menu items are available after the last displayed item.

3.1.3.2.9. Mutually exclusive attribute group markers

√ When a group of menu items represents a mutually exclusive (one of many) group of attributes (e.g., a font), then use a mutually exclusive attribute group marker to indicate the attribute currently in force (see Table 3-4. See Section 4.1.1 for more information on mutually exclusive attribute groups.



Type of Menu Choice	Apple	Motif	Windows
One of Many	◆		•
N of Many	√		√

Table 3-4. Markers for different choice groups.

√ In Apple, use the diamond to mark the current choice in a mutually exclusive attribute group. In Motif use a diamond as well. In Windows, the • is required for Ameritech applications.

Note: In Apple and Windows the symbol often used for mutually exclusive attribute groups is a check mark. The way that choice groups (i.e., mutually exclusive and non-mutually exclusive groups) work in current Apple and Windows applications this creates confusion among users; thus the need for two symbols

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.3.2.10. Non-mutually exclusive attribute group markers

√ When a group of menu items represent a non-mutually exclusive (n of many) group of attributes (e.g., text styles), then use a non-mutually exclusive attribute group marker to indicate the attributes currently in force (see Table 3-4). See Section 4.1.2 for more information on non-mutually exclusive attribute groups.

√ The symbol used for marking the current choice in a non-mutually exclusive attribute group in Apple and Windows is a check mark. In Motif the symbol is a diamond. See the previous section for more information.

More reading

Apple HIG, p. 65-73, 87-88.
OSF/MSG, § 3.3.2, 3.3.3, 4.1, 4.2.1.4, 7-13, 7-15, 7-96.
Windows ADG, § 5.2.2.1
IBM SAA CUA AIDR, p. 36, 138-142, 148-152

3.1.3.3. Optional elements

3.1.3.3.1. Accelerators

Accelerators provide a way to access menu elements from the keyboard without posting (i.e., displaying) the menu. See Section 3.4.2 for accelerators for standard menu items .

+ Use of keyboard accelerators is *strongly* encouraged.

√ Keyboard accelerators must appear in a left-justified column to the right of the menu item (see Figure 3-17b). The accelerator column must begin at the first tab position after the longest menu item (i.e., at least five spaces).

√ Menu items that have cascaded menus must not have accelerators.

√ The names used for the keys must match exactly how they are written on the keyboard. That is, in Windows use CTRL rather than CONTROL or ^. In Windows, do not use graphical representations of the keys. *Note:* This is more restrictive than the Windows ADG.

+ For frequently used items, provide an accelerator. It may be one function key or a combination of keys. When two or more keys must be pressed at the same time, use the system standard indication in the

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

menu item ("+" for Windows and Motif) to show multiple simultaneous key presses.

+ Accelerators should not require users to press more than three keys simultaneously.

+ Accelerators should have some associative value to the command. For instance, CTRL+F might be used for the `Find...` command.

More reading

Apple HIG, p. 72-73

OSF/MSG, § 3.3.2, 4.2.3.6

Windows ADG, 5.3.2

IBM SAA CUA AIDR, p. 125-130, 235-237

3.1.3.4. Properties

3.1.3.4.1. Enabling/disabling

Commands that are available to the user are . Commands that are not available are **disabled** (Apple uses the word "dimmed"). Enabled commands provide visual feedback (i.e., reverse video) when the user moves the cursor or pointer over them; disabled commands do not provide such feedback. (See Figure 3-19.)

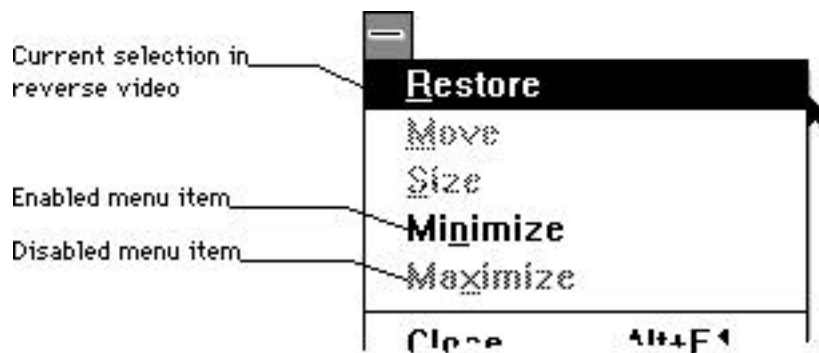


Figure 3-19. Examples of enabled and disabled menu items .

√ Commands must be enabled (1) when they apply to an object under focus (e.g., the menu item `Cut` is only enabled when an object is selected) or (2) when they are part of the application environment (e.g., `Open...`). Commands must be disabled when they do not apply to any object or the application environment.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 65
OSF/MSG, § 4.3.1.6
Windows ADG, §5.2.2.3
IBM SAA CUA AIDR, p. 262

3.1.3.5.

3.1.3.5.1. *Mouse pointer*

Picking an enabled menu item requires the user to move the pointer to a menu bar item and press the mouse button. This action drops down a menu. To pick an item, the user moves the pointer down the menu to the desired item and releases the mouse button. The menu goes away with no action if the user releases the mouse button before moving the pointer down the menu or by moving the pointer off the menu entirely and releasing the mouse button.

Exception: If the user moves the cursor off the menu the menu is “torn off” and its items are available to the user on a floating palette (Windows calls floating palettes, “” menus).

Motif and Windows provide a second method of picking an item from a menu. If the user clicks on the menu bar item the menu is posted. The user picks an enabled item by moving the pointer down the menu to the desired item and clicking the mouse button. The menu goes away with no action if the user clicks the mouse button off the menu area.

3.1.3.5.2. *Keyboard*

In Windows and Motif the user can post a menu by pressing the ALT and the mnemonic of the menu bar item. Once the menu is posted the up and down arrows on the keyboard move the cursor in the menu. Alternatively, the user can press the single letter mnemonic of a menu item that is on the posted menu.

√ If the cursor is at the top of the menu and the user presses the up arrow, the cursor must wrap to the last item on the menu. If the cursor is at the bottom of the menu, and the user presses the down arrow the cursor must wrap to the first item of the menu.

As of now, there are no accepted standards for controlling the Apple from the keyboard, although some applications (e.g., Microsoft Word 4.0) control the menu bar from the keyboard. Specific recommendations will be forthcoming in a future release. If you are considering allowing control of menus from the keyboard in Apple, follow the model of

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Microsoft Word 4.0. It is recommended that the F10 key be used as the key to activate the menu bar in Apple.

More reading

Apple HIG, p. 66
OSF/MSG, § 2.3
Windows ADG, § 5.3
IBM SAA CUA AIDR, p. 39-41, 138-142, 148-152, 193-194

3.1.4. Other standard controls

Each window manager provides a set of controls that can be used in the client area of the application. Each major control is described briefly below. More detail about the specifics of these controls can be found in Section 4 and in the respective style guides.

3.1.4.1. Command Buttons

Command buttons activate operations. They contain labels that indicate the action of the button. See Section 4.2.3 for more information on when to use command buttons. Motif and CUA refer to command buttons as "", Apple simply calls them "".

More reading

Apple HIG, p. 56.
OSF/MSG, p. 7-90.
Windows ADG, § 6.1, 7.3
IBM SAA CUA AIDR, p. 195-197.

3.1.4.2. Radio buttons

Radio buttons are presented in groups that allow for mutually exclusive settings of attributes. That is, if one of the radio buttons is selected all other radio buttons in the group are automatically deselected. Windows calls radio buttons "". See Section 4.1.1 for more information on when to use radio buttons.

More reading

Apple HIG, p. 57.
OSF/MSG, p. 7-96.
Windows ADG, § 6.1.2.
IBM SAA CUA AIDR, p. 201-202.

3.1.4.3. Check boxes

Check boxes allow for setting non-mutually exclusive attributes. Often check boxes occur in groups, any number can be selected at any time.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Motif calls check boxes “”. See Section 4.1.2 for more information on when to use check boxes.

More reading

Apple HIG, p. 56.
OSF/MSG, p. 7-15.
Windows ADG, § 6.2
IBM SAA CUA AIDR, p. 37-38.

3.1.4.4. Text boxes

Text boxes allow the user to enter free-form text. Motif calls text boxes “text”; CUA calls text fields “”. See Section 4.3 for more information on text boxes and data forms.

More reading

Apple HIG, p. 107, 120-121.
OSF/MSG, p. 7-120.
Windows ADG, § 6.4
IBM SAA CUA AIDR, p. 88-90.

3.1.4.5. List boxes

List boxes contain a set of items that the user can select or navigate through. List boxes can support either mutually exclusive selection or non-mutually exclusive selection from the list. See Section 4.1 for more information on the various uses of list boxes.

More reading

Apple HIG, p. 107, 116-117.
OSF/MSG, p. 7-59.
Windows ADG, § 6.3
IBM SAA CUA AIDR, p. 132-133.

There are three variations on lists: combination boxes, drop-down list, and drop-down combination box.

3.1.4.5.1. Combination boxes

Combination boxes bring together the functions of a list box with a text box. The user can either type into the text box at the top of the control or select an item from a list. Apple and Motif do not explicitly support this control. Using a text box and a list box one could design a control in Apple and Motif to perform the functions of the combination box.

More reading

IBM SAA CUA AIDR, p. 49-50.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.1.4.5.2. Drop-down list box

Drop-down list boxes allow the user to select an item from a list of items. The drop-down list box is distinguished from the list box in that the drop-down list box contains an arrow to the right of the control that, when selected, displays a list box. Apple and Motif do not explicitly support this control, however, many commercial applications have implemented this control. It is ideal when screen space is in high demand.

More reading

IBM SAA CUA AIDR, p. 84-85.
Windows ADG, § 6.4.1.1

3.1.4.5.3. Drop-down combination box

The **drop-down combination box** has the property of a list being available on demand, like the drop-down list, but it also allows users to type in entries, like the combination box. Apple and Motif do not explicitly support this control, although one can be designed (see Inside Macintosh, Vol. VI p. 2-37).

More reading

IBM SAA CUA AIDR, p. 82-83.
Windows ADG, § 6.4.1.2

3.1.4.6. Pop-up menu⁶

Pop-up menus contain sets of mutually exclusive or non-mutually exclusive options. For instance, a color palette is a set of choices that could be contained on a pop-up menu. See Section 4.1.1 for more information on pop-up menus, and Section 4.2 for information on commands. Motif has a control called "popup menus"; but the pop-up menus referred to here correspond more to the "" control. The difference is , as described in Section 4.2, that option menus "provide a means of selecting from a set of choices while taking up very little space", while popup menus in Motif can be used for commands.

+ In general, unless there is a strong reason to use pop-up menus for commands their use should be restricted to presenting sets of choices. Apple restricts the use of pop-up menus to sets of choices. Both Motif and CUA allow the use of commands on pop-up menus; thus, this recommendation restricts both the Motif and CUA guideline.

⁶ Windows uses the term "pop-up menu" in a much different way. They appear as floating menus when requested by the user (see Windows ADG, § 5.1.2).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 88-90.
OSF/MSG, § 4.2.3.
IBM SAA CUA AIDR, p. 184-185.

3.1.4.7. Slider

A **slider** is a graphical control that allows the user to set a value in a range of possible values by graphically moving a slider arm. See Section 4.1.1 for more information on using sliders. Apple calls sliders "▬"; Motif uses the term "▬".

More reading

Apple HIG, p. 57
OSF/MSG, p. 7-59
Windows ADG, § 6.6
IBM SAA CUA AIDR, p. 242-243

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.2. Navigation and Selection

The principal ways in which the user interacts with the application are with a pointing device, usually a mouse, and the keyboard. The mouse controls a screen cursor used to select objects and fix the location of other cursors, such as a text editing cursor. Keyboard navigation uses the TAB, , and COMMAND keys to shift the focus, make selections and execute commands. The balance of this section discusses cursor and keyboard navigation.

3.2.1. Cursor navigation

Cursors serve (1) to show the location of the operation (location cursor) and (2) to point to certain controls on the display (position cursor). The location cursor is usually (but not always) the I-beam cursor in text processing or data entry applications. The pointing cursor is often the left-pointing arrow. The **pointing cursor** is used to make selections from menus, click on buttons, resize windows, position the **location cursor**, etc. That is, the user moves the pointing cursor and clicks and the location cursor is moved. This is seen most often in text and field editing.

A cursor indicates the position of the user's operation and the system mode on the display. Table 3-5 shows several recommended cursor shapes and the states or modes to which each applies.

+ Unless absolutely needed, do not create special . Use the cursor shapes provided in the style guides.

+ An application should only use as many cursor shapes as it needs to keep the user completely informed about the location of the cursor, current mode, and current activity; too many shapes unnecessarily confuse users.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES













Name	Pointer Shape	Used For
Left-pointing arrow		Object selection and activation.
Right-pointing arrow		Indicating that there is a pending menu action (Motif) or in Apple it is used for selection in some instances. In Windows it is used for selecting lines, rows or cells.
I-beam		Changing location of text insertion cursor and performing actions on text.
(Motif)		Indicating when pointer is outside of any application area.
Move		Indicating a window is being moved or able to be moved. As appropriate, this pointer can be decomposed and used with only one line at a time (e.g., ).
Resize (Motif)		Indicating position and direction of a resize; typically used for windows.
Hourglass		Indicating that an action is in progress in the area, and that the pointer has no effect. Use the (Apple, Motif) or the beachball cursor (Apple).
Sighting, cross hair		Making fine position selections.
Plus sign		Selecting fields in an array (e.g., in a spreadsheet application).
Caution (Motif) Do-not (CUA)		Indicating that the target object is not valid for a direct manipulation operation.
Question		Requesting an input position or component from the user, usually for . Note that the cursor has a pointer at the top; this is the cursor's hot spot.

Table 3-5. *Pointer shapes.*

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 93-97.
OSF/MSG, § 2.2, 2.3.1, 3.1.1 through 3.1.5.
Windows ADG, § 3.6.1.1.1
IBM SAA CUA AIDR, p. 63-64, 175-179, 228-231.

3.2.1.1. Location cursor

√ Location cursors must be visually distinctive against all backgrounds, and must be easy to locate. For example, do not use a yellow cursor on a white background (see Section 3.3.1).

√ Location cursors never obscure other characters.

√ Generally, users set the blink rate for a location cursor through window management functions. But, if the application sets the blink rate, make the blink rate two to three hertz with a 50% duty cycle (i.e., ON half the time, off half the time).

√ The height of an I-beam location cursor must be the same height as the adjacent text.

3.2.1.2. Pointing cursor

√ Pointing cursors:

- maintain their size across all screen locations and during movement,
- may obscure screen objects,
- must never blink,
- must relate directly and rapidly (< 100 msec) to the movement of the pointing device, and
- never move without input from the user.

3.2.2. Keyboard navigation

Standard keyboard operation and selection is discussed when covering usage of controls, and is therefore not elaborated on here. For instance, keyboard selection of menu items is discussed in Section 3.1.3.5.2.

More reading

Apple HIG, p. 98-105.
OSF/MSG, § 2.3.2, 3.1.6, p. 7-73.
Windows ADG, §3.3.2
IBM SAA CUA AIDR, p. 125-130.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.2.3. Focus

Focus is the control in the window that will be next affected by the next input.

With the mouse, the focus is moved when the mouse is clicked on a control. For the keyboard, focus is moved to the next item in the tab order when the user presses TAB or when the user presses ALT+MNEMONIC for access to a random control.

+ The tab order should usually be from the top left to the bottom right.

+ The control receiving focus should give feedback to the user according to its type. The focus is indicated by an insertion point (e.g., an I-beam cursor for text) or highlighting for text, or a dotted box around the control's label.

The standards covering focus are detailed and should be reviewed in the style guide.

More reading

OSF/MSG, § 2.3
Windows ADG, § 3.2, 3.3
IBM SAA CUA AIDR, p. 122-130.

3.2.4. Selection

Various controls allow users to select either a single item or a group of items. For instance, a list box might support the user only selecting one item, items, or items.

Since these schemes are relatively homogeneous across platforms and built into the navigation of the controls, they are only briefly introduced here; see the respective style guides for exact and mouse-based selection techniques.

√ Do not use selection methods that are non-standard for your platform.

+ Allow users to make selections from the keyboard in addition to the mouse. One strong test of your application is to disconnect the mouse and see if the entire application is reasonably usable without the mouse. If not, work on areas where improvement is needed.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.2.4.1. Single selection

Only one object can be selected at any one time. For example, the object the user is pointing at (or where the cursor is located) becomes selected. Any previously selected object(s) is deselected.

This is similar to the one of many choices referred to in Section 4.1.1.

3.2.4.2. Multiple selection

The user can have any number of objects selected at any one time. For example, the user holds the mouse button and drags the pointer over a set of objects. All of the objects become selected. The user may extend the selection, typically by pressing SHIFT (Apple) or CTRL (Windows) and selecting another item or range.

+ Use of multiple contiguous and multiple discontinuous selection is strongly encouraged.

This is similar to the N of many choices referred to in Section 4.1.2.

More reading

Apple HIG, p. 106-117.

OSF/MSG, § 3.

Windows ADG, 3.1, 6.3.1, 6.3.2

IBM SAA CUA AIDR, p. 91-93, 159-161, 228-231, 240.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.3. Other topics

3.3.1. Use of color

3.3.1.1. Standard uses of color

Use color to add meaning; while color adds to the aesthetics of the interface, this is secondary.

Use color to:

- discriminate between different areas,
- show screen elements that are functionally related,
- show relationships among screen elements, and
- identify crucial features.

3.3.1.2. General design principles

+ Design applications in black and white; only when the application is finished should color be added.

Note: Some users will not have color monitors, 8% of the population is color deficient, and, under uncertain lighting, colors will be difficult for the eye to distinguish.

+ It is strongly suggested that color only be used to supplement other highlighting techniques (see Section 5.1.5.). There should always be font, shapes, locations, or patterns that also distinguish screen elements in addition to color.

Too much color can have the effect of making something more difficult, rather than more simple to use. User studies indicate that users can only effectively follow four colors on a screen at once, and no more than seven over the entire application.

+ Allow users to change the colors, and to change default color codings that is used as default. A future release of this document will contain recommended color values.

+ Use of color in the client area depends on the application.

△ Note that the meaning of a color is task dependent. The color blue in a figure does not mean the same thing medical application as it does in a map application. Know the meanings of colors for the domain(s) that your application covers. Tables 3-6 and 3-7 show recommended colors for certain interface objects and colors to use for certain types of coding.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Object	Recommended color
Background	Light gray or white
Text	Black
Borders	Black
Controls	Black
Pointer	Black with white outline

Table 3-6. Recommended colors for interface objects.

Color	Indicates	Use for
Red and orange	Stop, error, failure	Attracting user's attention
Blue and green	Go, ready	Telling user of normal operation or process is completed and ready for action.
Light blue		Making something unobtrusive, like grid lines.

Table 3-7. recommended colors for certain actions.

More reading

Apple HIG, p. 30-35.
Shneiderman, p. 336-352.
Smith, § 2.6.
IBM SAA CUA GUID, p. 137.

3.3.2. Use of sound

3.3.2.1. Standard uses of sound

Use sound to:

- make the user aware of the state of the computer or application,
- alert the user when something unexpected happens, and
- notify the user when immediate attention is required.

3.3.2.2. General design principles

√ and error dialog boxes must be accompanied by a sound.

√ Use sound consistently. If one signal is used for a certain activity or error condition, do not use that tone for other situations.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Do not use more than two notes in any given alert; avoid use of tunes or jingles.

+ Always allow the user to adjust volume or turn the sound off altogether.

+ Be careful not to overuse sound or the sound will be ignored or be annoying.

+ Make sounds effective, but subtle. Always test sounds with users over extended trial periods and not just casually.

+ You may use as many as six different tones in a single application, but make them significantly different so that users can discriminate among them.

+ Signal frequencies should be between 500 and 1000 hz.

Δ Remember that computers are used in groups; sounds affect users other than the one using the application. Avoid the use of loud sounds.

More reading

Apple HIG, p. 36-37.

Smith, § 2.6.

IBM SAA CUA AIDR, p. 34.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4. The Ameritech Signature

3.4.1. Use of logo, trademarks, and proprietary markings

{To be completed in future release}

3.4.2. Ameritech

The Ameritech standard menus provide a consistent starting point for all users of the applications. There are certain practices (as presented in the respective style guides and from observing several applications) and philosophies that determine which items should be made standard. One philosophical disagreement that is often heard surrounds the use of the term "File" instead of a more application-specific term, e.g., "Account", for a menu title. Since the term "File" is in widespread use today and most users quickly learn the contents of the file menu, it must be used in place of the application-specific term. It cannot be guaranteed that every application will have a good term for these functions; additionally, virtually all applications will be retrieving and storing files, printing, quitting, etc. These all group well under "File", but do not group as well under "Account". Lastly, there is a certain amount of expectation that users who have used other products bring to using any application - the functions and contents of the "File" menu being part of that expectation. Using an application-specific term violates that expectation without adding a lot.

√ The Ameritech standard menus are: Control, File, Edit, Help, and if the application can have more than one primary window, Window. All but the Control menu, must appear on the menu bar of all Ameritech applications. Apple does not have the Control menu and it resides in the title bar area for Motif and Windows. Each menu is presented below along with its required menu items.

3.4.2.1. for primary windows

Description

The **control menu** (Motif calls it the "window menu") provides the user with choices that affect the window and other operating system-oriented actions. The actions described here correspond to similar actions described in Section 3.1.1.3; refer to Section 3.1.1.3 for precise behavior of these actions. Figure 3-20 shows an example.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES



Figure 3-20. An example of a control menu in Windows.

√ The control menu itself is required, but not all items on the control menu are required; required items are marked in Table 3-8 in the Action column. Also, if appropriate, you may include items not described here, but they must be placed on the menu below the items presented here. Last, Windows only requires the **C**lose and **S**witch To menu items to have accelerators; the requirements here are more strict than those for Windows.

Table 3-8 shows the accelerators and mnemonics for the Control menu.

Action	Accelerators		Mnemonics	
	Windows	Motif	Windows	Motif
Control menu	SHIFT+ESC	SHIFT+ESC	Spacebar	Spacebar
√ R estore	ALT+F5	ALT+F5	R	R
√ M ove	ALT+F7	ALT+F7	M	M
√ S ize	ALT+F8	ALT+F8	S	S
√ M inimize	ALT+F9	ALT+F9	N	N
√ M aximize	ALT+F10	ALT+F10	X	X
+ L ower	NA	ALT+F3	na	L
√ C lose	ALT+F4	ALT+F4	C	C
+ S witch to...	CTRL+ESC	NA	W	na
+ N ext	ALT+F6	NA	T	na

Table 3-8. accelerators and mnemonics for the control menu.

LEGEND: √ Required +Recommended Δ Good Practice

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Placement	Place control menu at the first position on the title bar.
<i>More reading</i>	OSF/MSG, § 5.3.7. Windows ADG, § 5.4.1. IBM SAA CUA AIDR, p. 253-254.
3.4.2.1.1. Restore	
Description	Restore returns a minimized or maximized window to its previous size and location. When the window is neither minimized or maximized, disable Restore .
Placement	Place Restore at the first position of the control menu.
3.4.2.1.2. Move	
Description	Move enables the user to move a window to a different location on the display. This option allows users to Move the window without using the mouse.
Placement	Place Move in the second position of the control menu, immediately following Restore .
3.4.2.1.3. Size	
Description	Size allows the user to change the height and width of the window. This option allows users to resize the window without using the mouse.
Placement	Place Size in the third position of the control menu, immediately following Move .
3.4.2.1.4. Minimize	
Description	Minimize removes a primary window and all its secondary windows from the workplace, and replaces them with an application-defined icon in the workplace or desktop. When the window is minimized, disable Minimize .
	<p>√ Use the term “minimize” instead of hide or iconize. Some applications use “hide” or “iconize” instead of minimize; do not do either of these terms.</p>
Placement	Place Minimize in the fourth position of the control menu, immediately following Size .

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.1.5. *Maximize*

Description **Maximize** enlarges a window to its largest size or to the size of the workplace, whichever is *smaller*. **Maximize** must be enabled when the window can be resized, but is not currently maximized. When the window is as large as it can be, disable **Maximize**.

Placement Place **Maximize** in the fifth position of the control menu, immediately following **Minimize**. If not using **Lower**, place a separator after **Maximize**.

3.4.2.1.6. *Lower*

Description **Lower** moves a window to the bottom of the window hierarchy in Motif.

Placement Place **Lower** in the sixth position of the control menu, immediately following **Maximize**. Place a separator after **Lower**.

3.4.2.1.7. *Close*

Description **Close** removes a window and all secondary windows. If the user has not saved the contents of the window and selects **Close**, prompt the user with the standard save changes dialog box (see Section 3.4.3.3) before closing.

Placement Place **Close** immediately following the separator after **Maximize** or **Lower** (if used). Place a separator after **Close**.

3.4.2.1.8. *Switch to...*

Description **switch to...** displays a window containing all active applications in the Windows environment. The user can move to any open application by selecting an application name from this list.

Placement Place **Switch to...** immediately following the separator after **Close**. Place a separator after **Switch to...**

3.4.2.1.9. *Next*

Description **Next** opens the next application in the application list in Windows.

Placement Place **Next** immediately following **Switch to...**

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.1.10. Other application menu items

Other application-oriented file menu items may be added after a separator following `Next`, but they should be limited to commands relating to providing keyboard control over the active window.

3.4.2.2. for secondary windows

√ The control menu for secondary windows must use, in the following order, `Restore`, `Move`, `Size`, and `Close` (see descriptions in previous Section - 3.4.2.1).

Placement Place control menu at the first position on the title bar.

More reading OSF/MSG, § 5.3.7.
Windows ADG, § 5.4.1
IBM SAA CUA AIDR, p. 253-254.

3.4.2.3. File

Description The File menu title on the menu bar provides the user with application-oriented commands that deal with file input, output, creation, printing, and ending the application. Additional items may be added if they deal with managing files in the user's environment.

Table 3-9 shows the accelerators and mnemonics for the File menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Action	Accelerators			Mnemonics	
	Apple	Windows	Motif	Windows	Motif
File menu	NA	NA	NA	F	F
New	⌘N	NA	NA	N	N
Open...	⌘O	CTRL+F12	CTRL+F12	O	O
Save	⌘S	SHIFT+F12	SHIFT+F12	S	S
Save as...	None ⁷	F12	F12	A	A
Close (Apple Only)	⌘W	NA	NA	NA	NA
Print...	⌘P	CTRL+ SHIFT+F12	CTRL+ SHIFT+F12	P	P
Exit/Quit	⌘Q	ALT+F4	ALT+F4	X	E

Table 3-9. Accelerators and mnemonics for the File menu.

Placement Place File at the first position on the menu bar.

More reading Apple HIG, p. 75-80.
Windows ADG, § 5.4.2.
OSF/MSG, § 4.2.1.4.1.
IBM SAA CUA AIDR, p. 96-99.

3.4.2.3.1. New

Description **New** creates a new file of the file type used by the application. For example, new creates a new document for a word processor, or a new record in a database application. If your application can produce several different kinds of file types, e.g., graph, document, record, etc., then you should list new with an ellipsis (i.e., *New. . .*), and present the user with a list of file types from which to choose. If selection of new will cause the loss of data in the current application, then present a standard "Save changes dialog box" (see Section 3.4.3.3). **New** is disabled when the maximum number of documents allowed by the application is already open.

Placement **New** must be in the first menu position under the File menu.

⁷ Do not use ⌘A. The most recent user interface guidelines from Apple have changed ⌘A to mean "Select All".

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.3.2. *Open*

Description **Open** retrieves an existing file, of the file type used by the application, by routing the user to the File open dialog box (see Section 3.4.3.1). If the application has several different file types then present the user with the File New dialog box in Section 3.4.3.

Placement Place **Open** at the second menu position under the File menu, immediately below **New**.

3.4.2.3.3. *Save*

Description **Save** writes the data associated with the active primary window without removing or changing the contents of the client area. If the current file has no user-defined name, **Save** routes the user to the save as dialog box (see Section 3.4.3.2). Do not close the window after the user has saved the file.

Placement Place **Save** at the third menu position under the File menu, immediately below **Open**. You may, optionally, place a separator between **Open** and **Save**.

3.4.2.3.4. *Save as...*

Description **Save as...** routes the user to the "save as dialog box" (see Section 3.4.3.2) where the user can save the active primary file under a different name or different file type without removing or changing the contents of the client area. Changes are not made on the original file. If the user tries to save the document with an existing name, a secondary dialog box should alert the user to the possible loss of data, and allow the user to enter a different name. Do not close the window after the user has saved the file. **Save as** should allow the user to save the data in multiple formats.

Placement Place **Save as** at the fourth menu position under the File menu, immediately below **Save**.

3.4.2.3.5. *Close*

Description **Close** terminates the active primary file in Apple. If the current file has no name or if the file has been changed since it was last saved, **close** routes the user to the appropriate dialog box for saving (see Section 3.4.3.2). **Close** also removes any secondary windows associated with the active primary file. Any other open primary windows are not closed.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

For instance, in a word processing program the user might have three different open files. If the user selects close, only close the active primary window, i.e., close only one file. Of the remaining files, the window that was most recently active, is made active again.

Placement Place `Close` at the fifth menu position under the File menu, immediately below `Save as`. Place a separator after `Close`.

3.4.2.3.6. *Print*

Description **Print** prints the currently open file.

Placement Place `Print` at the sixth menu position under the File menu, immediately after the separator following `Close`.

3.4.2.3.7. *Other application menu items*

Other application-oriented file menu items may be added between the separator and `Exit/Quit` (e.g., `Import...`).

3.4.2.3.8. *Exit/Quit*

Description **Exit/Quit** ends the current application and closes all associated files. All primary windows must be closed (see description of close above), including prompting users to save files that have not been saved. Apple uses `Quit`. Windows and Motif use `Exit`.

Placement Place `Exit/Quit` at the last menu position under the File menu. Place a separator before `Exit` or `Quit`.

3.4.2.4. **Edit**

Description The Edit menu title on the menu bar provides the user with undo capability and options related to the clipboard.

Table 3-10 shows the accelerators and mnemonics for the Edit menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Action	Accelerators			Mnemonics	
	Apple	Windows	Motif	Windows	Motif
Edit menu	NA	NA	NA	E	E
Undo	⌘Z	CTRL+Z & ALT+ BACKSPACE	ALT+ BACKSPACE	U	U
Redo	⌘Z	CTRL+Z & ALT+ BACKSPACE	SHIFT+ALT+ BACKSPACE	R	R
Cut	⌘X	CTRL+Z & SHIFT+DEL	SHIFT+DEL	T	T
Copy	⌘C	CTRL+C & CTRL+ INSERT	CTRL+ INSERT	C	C
Paste	⌘V	CTRL+V & SHIFT+ INSERT	SHIFT+ INSERT	P	P
Select all	⌘A	CTRL+/	CTRL+/	S	S
Clear	DELETE	DELETE	DELETE	E	E

Table 3-10. Accelerators and mnemonics for the Edit menu.

Placement Place Edit at the second position on the menu bar.

More reading Apple HIG, p. 80-84.
OSF/MSG, § 4.2.1.4.2.
Windows ADG, § 5.4.3.
IBM SAA CUA AIDR, p. 86-87.

3.4.2.4.1. Undo/Redo

Description **Undo** reverses the effect of the last applied operation on an object. Design the application so that any user action is immediately reversible including text deletion, formatting, and control settings. **Redo** reverses the effect of **Undo**. **Undo** and **Redo** labels should be dynamic, to indicate what is being undone or redone. For instance, if the action was a **Cut** then the **Undo** menu item should read **Undo cut**. **Undo** is disabled when there is nothing to undo, plus the label should be changed to read **Can't undo**.

Placement Place **Undo** at the first menu position under the Edit menu. Put a separator after **Undo/Redo**.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.4.2. *Cut*

Description	Cut removes the selection from the client area to the clipboard. The application determines what happens to the client area the object occupied. (That is, whether it remains blank and the space is compressed or not.) Cut is disabled when there are no objects selected.
Placement	Place Cut at the second menu position under the Edit menu, immediately following the separator after Undo .

3.4.2.4.3. *Copy*

Description	Copy duplicates the selected objects from the client area onto the clipboard without affecting the selection. Copy is disabled when there are no objects selected.
Placement	Place Copy at the third menu position under the Edit menu, immediately following Cut .

3.4.2.4.4. *Paste*

Description	Paste places the contents of the clipboard into the client area. If there is no object selected, the clipboard is pasted at the current location of the cursor in the client area. If an object is currently selected in the client area, Paste places the clipboard contents in the location of the selected object, replacing the selected object. Paste is disabled when there are no objects to paste.
Placement	Place Paste at the fourth menu position under the Edit menu, immediately following Copy . Place a separator after Paste .

3.4.2.4.5. *Select all*

Description	select all allows the user to select all of the objects in the client area.
Placement	Place Select all at the fifth menu position under the Edit menu, immediately following the separator after Paste .

3.4.2.4.6. *Clear*

Description	Clear removes the selection from the client area <i>without</i> placing it on the clipboard. Clear removes the selection, but does not compress the
-------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

space the selection occupied. `Clear` is disabled when there are no objects to clear.

Placement Place `Clear` at the sixth menu position under the Edit menu, immediately following `Select all`. Place a separator after `Clear`.

3.4.2.4.7. Other Edit menu items

Place all other application-oriented edit menu items after the separator following `clear`. Windows recommends that Find and Replace commands be added to the Edit menu.

3.4.2.5.

Description The window option on the menu bar provides the user with application-oriented commands that deal with window management *within* the application (in addition to outside the application for MDI applications). All primary windows and window-oriented functions (those below and other optional ones) should be listed in the window menu, as shown in Figure 3-21. A window menu is offered on many Macintosh and Windows application software. It is not discussed, however, in the Apple or Motif style guides.

Table 3-11 shows the accelerators and mnemonics for the window menu.

Action	Accelerators		Mnemonics	
	Windows	Motif	Windows	Motif
Window menu	NA	NA	W	W
Tile	SHIFT+ F4	SHIFT+F4	T	T
Overlap	SHIFT+ F5	SHIFT+F5	O	O
Open window	NA	NA	{number}	{number}

Table 3-11. Accelerators and mnemonics for the window menu.

Placement Place Window at the second to last position on the menu bar, immediately before Help.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Window	
T ile	Shift+F4
O verlap	Shift+F5
1 Accessories	
◆ 2 Main	

Figure 3-21. A standard window menu in Windows.

More reading

Windows ADG, § 5.5.2.
IBM SAA CUA AIDR, p. 283.

3.4.2.5.1. Tile

Description

Tile takes all primary windows and places them on the display in a non-overlapping fashion. For instance, if tile was selected from the window menu in an application with four primary windows, each window would be moved and resized by the application to occupy one quadrant of the user's screen. The active window should be placed in the upper left corner. Figure 3-22 shows the effects of the **T**ile command.

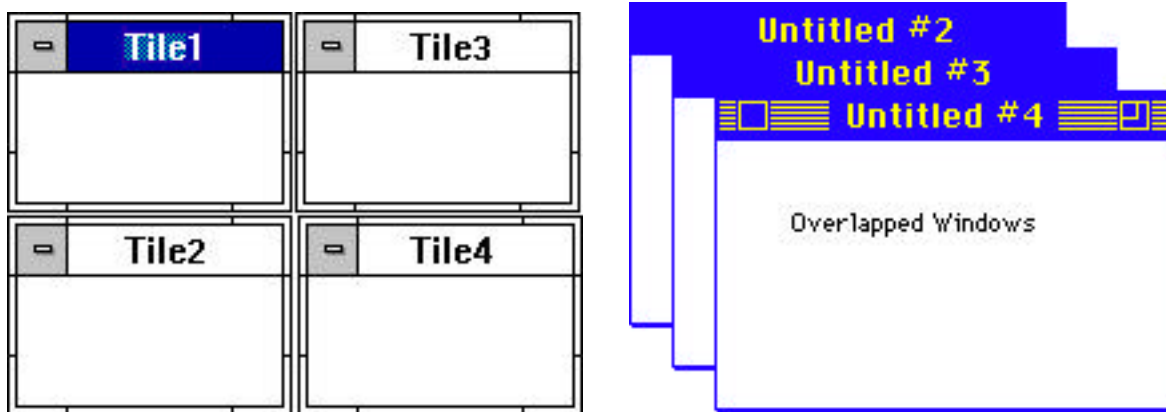


Figure 3-22. Examples of tiled (left) and overlapped (right) windows.

Placement

Place **T**ile at the first menu position under the window menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.5.2. *Overlap*

Description	Overlap takes all primary windows and places them one on top of another below and to the right of each other (see Figure 3-22). Each window should be made a uniform size. The active window should be the unobstructed window on the display.
Placement	Place Overlap at the second menu position under the Overlap should follow Overlap .

3.4.2.5.3. *Other window menu items*

Place all other application-oriented window menu items after a separator following **Overlap**.

3.4.2.5.4. *{List all open windows}*

Description	All primary windows for the application should be listed in alphabetic order by their name in the window menu. In Windows and Motif, precede each primary window with a dynamically assigned number (beginning with number "1") to be used as the mnemonic. For example, if the primary window named "Pulleys" was in the menu, it would be represented as " <u>1</u> . Pulleys".
Placement	Place the window names and their dynamically assigned consecutive numbers after the separator following overlap .

3.4.2.6. **Help**

Description	The Help menu on the menu bar provides the user with options for obtaining help on the application. The application can have other categories for help; these are recommended as the minimal set. Since the help system in Windows is well-evolved, and there are mimics of that help system in Apple, the help menu and structure recommended here will follow closely that described in Windows ADG. See Sections 5.2.3 and 5.3.4 for information on writing help text itself.
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Table 3-12 shows the accelerators and mnemonics for the help menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Action	Accelerators			Mnemonics	
	Apple	Windows	Motif	Windows	Motif
Help menu	NA	NA	NA	H	H
Contents	NA	F1	F1	C	C
Search for Help On	NA	NA	NA	S	S
Tutorial	NA	NA	NA	T	T
How to Use Help	NA	NA	NA	H	H
About <application name>	NA	NA	NA	A	A

Table 3-12. Accelerators and mnemonics for the Help menu.

Placement Help is always the last item on the menu bar, immediately following the next to last item.

More reading OSF/MSG, § 4.2.1.4.3.
Windows ADG, 5.4.4.
IBM SAA CUA AIDR, p. 108-109.

3.4.2.6.1. Contents

Description **Contents** opens the help system and displays a list of all the main topics in the application.

Placement Place **Contents** in the first position of the Help menu.

3.4.2.6.2. Search for Help On

Description **search for help on** opens the help window and allows the user to search among the help topics for one containing specific keywords.

Placement Place **Search for Help On** in the second position directly below **Contents** in the Help menu.

3.4.2.6.3. Tutorial

Description **Tutorial** is an optional item that is present if and only if the application has any on-line tutorials available.

Placement Place **Tutorial** in the third position directly below **Search for Help On** in the Help menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.2.6.4. *How to Use Help*

Description How to Use Help provides information on how to use the help system.

Placement Place How to Use Help in the fourth position directly below Tutorial in the Help menu.

3.4.2.6.5. *Other optional items*

Description There are a number of common optional items such as: Commands , Procedures , Keyboard, etc. Help should be tailored specifically to meet the needs of your application.

Placement Place optional items in the positions before About <application name> in the Help menu.

3.4.2.6.6. *About <application name>*

Description About <application name> displays a standard dialog box containing the name of the application, version number, copyright message, icon serial number, user's name, amount of available workspace in memory, and available storage space.

Placement Place About <application name> in the last position in the Help menu.

3.4.3. Ameritech standard dialog boxes

Certain actions result in dialog boxes that are common across all platforms and applications. The toolkits provide common dialog boxes for many of the actions described below. You should use those dialog boxes in your application, as listed below. In the following, you should look at all (or in some cases, the only) style guide for an example if one does not exist for your platform. For instance, Motif does not have a standard Print Setup dialog box. In that case, if you are developing a Motif application you should consult the Windows and Apple style guides for the minimum controls needed and the rough layout.

It is anticipated that this section will expand substantially with future releases of this document.

3.4.3.1. File open dialog

When the user issues a command to open a file, the user should see a dialog box that allows the user to enter the name or locate the file in the

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

directory and the various storage devices. There should be a `Help` button on this dialog box. Other elements are often necessary and helpful to the user. For instance, in pixel-based graphics applications the File open dialog box might have a preview area allowing the user to get a snapshot of the file before actually pulling it up..

Among the following, both the Motif and Windows examples are well-constructed. The Apple dialog box is somewhat dated.

See Apple HIG, p. 76.
OSF/MSG, p. 7-41.
Windows ADG, § 8.1.1.

3.4.3.2. File save as dialog

If the user must save the current document under a different name or in a different file format from the original saved version or if the user has never saved the file, present a save as dialog box, as shown in Figure 3-23.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

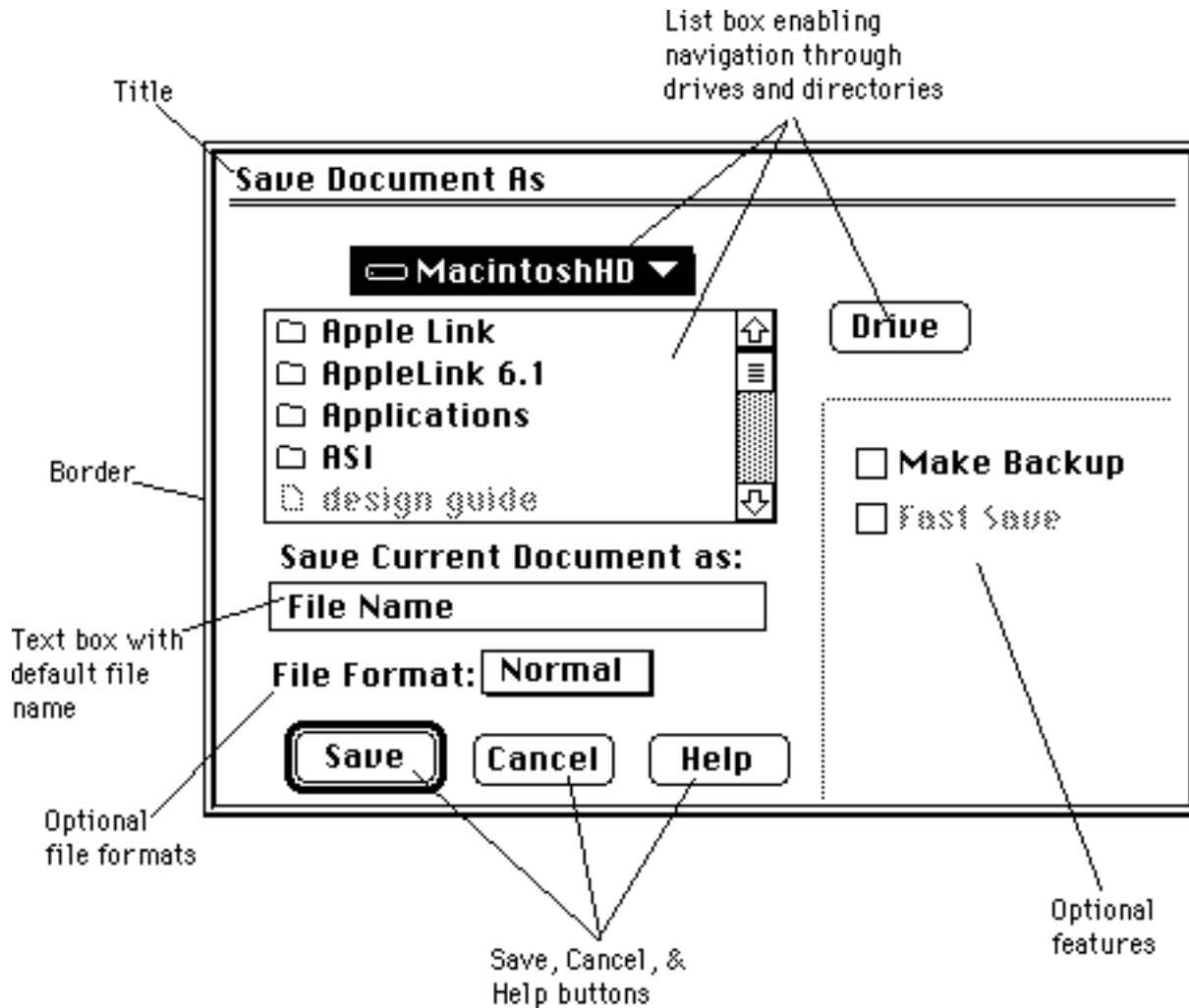


Figure 3-23. Save as dialog box.

See Apple HIG, p. 78.
Windows ADG, § 8.1.2.

3.4.3.3. Save changes dialog

If the user attempts to close a window before the contents of the window have been saved, prompt the user with the dialog box in Figure 3-24.

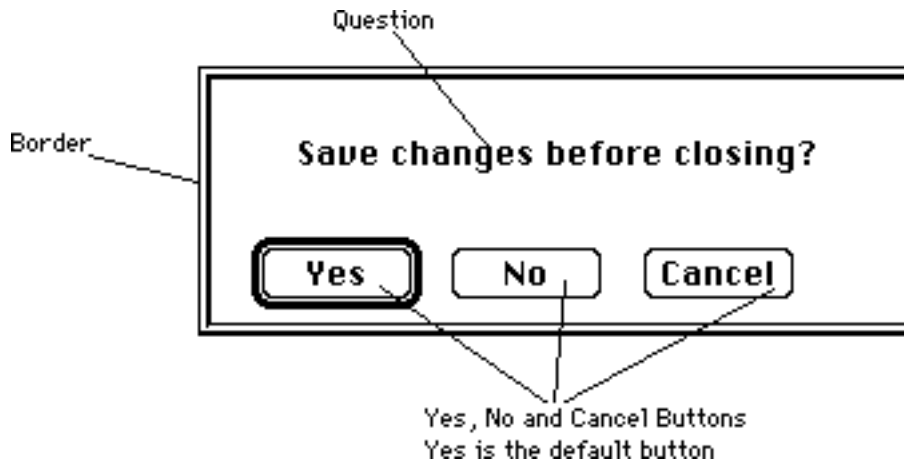


Figure 3-24. Save changes dialog box.

See Apple HIG, p. 77.

3.4.3.4. File new dialog

When you have an application that uses several different file types, then after the user selects the *New...* menu item from the *File* menu the *File New* dialog should appear. It contains a list box with the different document types used by the application and *OK*, *Cancel*, and (especially) *Help* buttons.

See Windows ADG, § 8.1.3.

3.4.3.5. Print dialogs

When your application has to send documents to the printer, use the print dialog. Usually this means when the user has selected *Print* from the *File* menu. The print dialog tells the user where the document will be printing, requests that the user select the pages to print, how many copies to print, and what print quality to use. There should also be options for the user to select a different printer (in Apple this is handled through the *Chooser*), and to select different options (see *Print setup* below). The Windows dialog box is the best of the two listed below.

See Windows ADG, § 8.2.1, 8.2.2
Apple HIG, p. 79.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.3.6. Printing

When the user issues the command for the print to take place, the user should see a dialog box with the elements in Figure 3-25.

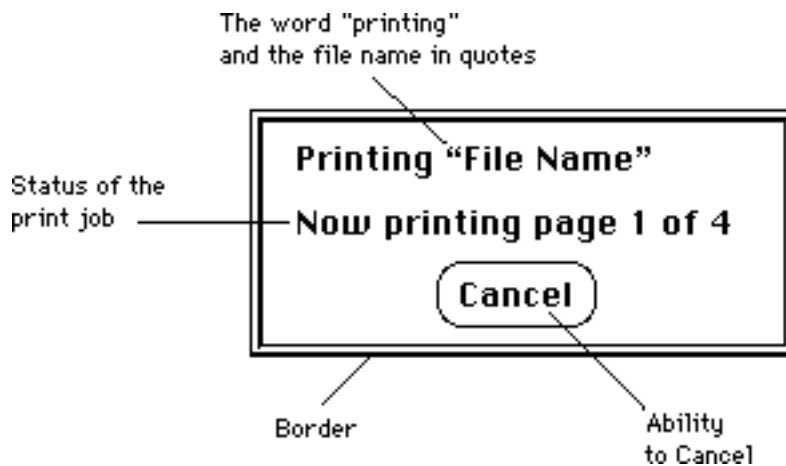


Figure 3-25. Printing dialog box.

The printing dialog box is modal; it is presented while the file is being printed or being spooled to the printer.

3.4.3.7. Printer error

When printing a variety of errors can occur, many of those errors and suggested message dialog boxes are presented in Windows ADG § 8.2.1.

3.4.3.8. Find and Replace Dialogs

Find and replace operations are common in many different types of applications. It is important to standardize those dialogs. Windows ADG, shows the minimal set of controls needed. For Find operations, the user must be able to enter a text string, specify the direction of the search (a "selection" radio button should be added to the set including "up" and "down") and set other search-limiting parameters. Depending on your application, you may want to use different search-limiting parameters. Also, allow users to search on non-printing characters like tabs (use "^t") and paragraph markers (use "^p").

See Windows ADG, § 8.3.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.3.9. Character properties Dialogs

In Section 4 there is discussion of when to put controls on the menu bar and when to put them in dialog boxes. If you determine that your application should use a dialog box for setting text properties, then you should follow the dialog boxes laid out in the Windows ADG for setting character properties. These are well designed.

See Windows ADG, § 8.4.

3.4.3.10. About <application name> dialogs

When posting the "About <application name>" dialog box after the user selects *About application name...*, it should contain the elements presented in Figure 8.11 in the Windows ADG.

See Windows ADG, § 8.6.

3.4.4. Ameritech color palettes

{To be completed in future release}

3.4.5. Common interface objects

This section presents some common fields and the standard (required) formats. The fields presented here will be researched and may change in the future as well as the fact that more will be added in future releases of this document.

3.4.5.1.

√ Abbreviated dates must be presented in MM-DD-YY format; example, 12-05-92.

√ When the date is written out completely it must be of the form DD Month YYYY; example, 5 December 1992.

3.4.5.2.

√ When displaying time in the 12 hour clock, use the following format HH:MM:SS XM Time Zone. Example: 4:15:22 PM CST.

√ When displaying time in the 24 hour clock, use the following format HH:MM:SS Time Zone. Example: 16:15:22 CST.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

3.4.5.3.

√ Use the following format to request and present telephone numbers
NXX-NXX-NNNN; Example, 908-758-2404.

3.4.5.4. Nine-digit ID numbers

√ Use the following format to request and present nine-digit numbers
(e.g., social security numbers):

Social Security Number:

3.4.5.5. Icons

{To be completed in future release}

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4. Guidelines for Obtaining Input from the User

Contents

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.1. Selecting from a Set

4.1.1. "One of many" Choices

Often the application restricts the user to choose one item from a group of possible items. These are "**one of many groups**" or **mutually exclusive choice groups**. For instance, in a paint application the user might have a variety of tools with which to paint - a brush, a pencil, a roller, and a spray can. Even though all of these tools are available, the user may only use one tool at a time. Some examples include a list of fonts in a word processor, or the settings of the sound level.

A special case of mutually exclusive choices is the **binary choice**. For instance, a switch for a printer is either on or off. Another example might include a choice between one object or another object; the user can be using either printer 1 or printer 2, but not both. There are some special considerations given to binary choices; these will be pointed out below as needed.

The choice of the correct interface form for mutually exclusive choice groups depends on the following criteria:

- (1) number of times used,
- (2) number of items in the group,
- (3) space available on the display,
- (4) frequency with which items are added or deleted from the group,
- (5) nature of the attributes themselves (i.e., discrete or continuous attributes)

When using a mutually exclusive choice, you should be most sensitive to the number of times the mutually exclusive choice group is used. For example, if the user is frequently changing the attribute values (e.g., in a paint application the user might be switching paint tools frequently), do not use a dialog box. This would be too tedious. It would be better, in this case, to use a pull-down menu or a continuously displayed icon panel.

Table 4-1 presents the selection criteria in the left column and the recommended interface form in the right column. If your criteria do not fit exactly into one of the boxes on the left column choose the one that is closest to the kind of items you have. Each of the interface forms and its variants is discussed in detail below.

If Mutually Exclusive Items are...	Then Use the Following Form...

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

<p><i>attributes or values</i> that</p> <ul style="list-style-type: none">• are selected frequently• are limited in number (2-8)• are best represented verbally, and• change rarely.	Attribute menu items
<p><i>attributes or values</i> that</p> <ul style="list-style-type: none">• are selected infrequently• are limited in number (2-8)• change infrequently, and• require little screen space.	Pop-up menu or Drop-down list
<p><i>attributes or values or objects</i> (e.g., file names) that</p> <ul style="list-style-type: none">• are potentially large in number• are best represented verbally, and• can change frequently.	Single-selection lists
<p><i>attributes or objects or values</i> that</p> <ul style="list-style-type: none">• are selected infrequently• are relatively few in number (8 or fewer)• have sufficient screen space available, and• do not change.	Radio buttons
<p><i>commands</i> that</p> <ul style="list-style-type: none">• are selected frequently, and• have only two conditions.	Toggled-menu items

Table 4-1. This table presents the conditions for selecting the proper interface form for mutually exclusive choices. (Table continued on next page...)

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

If Mutually Exclusive Items are...	Then Use the Following Form...
<i>commands</i> that <ul style="list-style-type: none">• are selected infrequently, and• have only two conditions.	Toggled-command buttons
<i>attributes</i> that <ul style="list-style-type: none">• are selected very frequently• are limited in number• are best represented graphically• are discrete in nature, and• change very rarely.	Icons, icon lists, or icon palettes
<i>attributes</i> or <i>values</i> that <ul style="list-style-type: none">• are best represented graphically, where other forms do not apply.	Specialized graphical control

Table 4-1 (cont'd). This table presents the conditions for selecting the proper interface form for mutually exclusive choices.

4.1.1.1. Interface Forms

4.1.1.1.1. Attribute menu items

Attribute menu items are attributes or values (e.g., 12 Point), not commands (e.g. Rotate). Attribute menu items are located on a pull-down menu. The user picks an item by selecting the pull-down menu and moving the pointer to the desired item and selecting it. The selection mark then moves (or remains, if the same item is picked) to the newly selected items. When selected, these attributes remain in force until they are de-selected in favor of another item.

4.1.1.1.1.1. Proper usage

- + Use an attribute menu when
 - options must be accessed frequently
 - the number of options in the group is small (i.e., no more than eight), and

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- the items in the group rarely change.

Comment. There is a tradeoff between the time it takes to find an item and the number of selections that must be made. While the optimal menu design for any application is task dependent, research on menu design indicates that the optimal number of items per menu group⁸ is between four and eight. Moreover, there is another practical consideration: As the number of menu items increases, the menu gets longer. It is strongly suggested that you limit the size of the menu such that the whole menu can be displayed entirely without scrolling.

The frequent (and excusable) exception to the numerical limits are the `Font` and `Size` menus used in most text editors. A system may have available hundreds of fonts that the user might wish to access regularly. Most users will have between 5 and 20 fonts installed; these should be listed on an attribute menu. Putting the list of fonts in a scrolling list on a dialog box requires users to work more than do attribute menus. (However, a scrolling list may be acceptable if the fonts are not regularly added and/or deleted; see below.)

The symbol used to mark the selection on the menu itself differs across platforms and is discussed in Section 3.1.3.2.

Figure 4-1 shows an example of the attribute menu. The `Font` menu is used to pick the active font in a text editor. In this case, `Athens` has a diamond next to it, meaning the `Athens` font is currently in use. If the user next picked `London` the diamond would move to `London`, and the `London` font would be applied to the objects in force at the insertion point.

⁸ A “group” is any set of menu items. A group on a pull-down menu is anything between two separators or the whole list if there are no separators on the pull-down menu. Choice groups must be offset by separators.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES



Figure 4-1. An attribute menu list is used to set one item in force. In this case, the Athens font is the one selected.

4.1.1.1.1.2. Variations

One variation of the attribute menu is to put the attribute choices on a cascaded menu. For instance, many text editors use `Text` as the menu title. Within the `Text` menu they list `Font`, `Size`, and `Style` as three menu items, each with a cascaded menu. In general, the use of cascaded menus for making mutually exclusive choices is acceptable with the following three caveats. First, *do not* go beyond one level of a cascaded menu; it is too hard for users to keep track of all the settings or to undo a mistaken setting. For instance, with cascaded menus, if the user wants to see the font size, she or he has to pull down the menu and make another selection before she or he can find the current font size. Second, cascaded menu items must be attributes; do not put commands on cascaded menus. Third, consider using cascaded menus only when it makes sense to use a pull-down menu, *and* the menu bar is getting crowded with other entries.

4.1.1.1.1.3. Misusage

√ Attribute menu items must not be used as commands. If the user will frequently access a set of choices, consider using a palette instead of an attribute menu.

More Reading

Apple HIG, p. 67-69.
OSF/MSG, § 4.2.3.
Windows ADG, § 5.1.1., 5.1.3., 5.2.2.1, 5.3.1.1.3.
Kobara, p. 112-116, 152-154.
OL/ASG, p. 293-294.
IBM SAA CUA AIDR, p. 233-234.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.1.1.1.2. Pop-up menu

Pop-up menu and a **drop-down lists** allow the user to select one of many choices.⁹ Figure 4-2 shows an example of a pop-up menu in the Macintosh user interface; pop-up menus look different from other similar-looking objects - buttons and list boxes - because they have a drop shadow and the cursor does not change when the mouse passes over it. When the user selects the box, a list of valid items appears and the user can make a selection by moving the mouse to the appropriate choice. The currently selected item is marked with a diamond, just as for attribute menus. Figure 4-3 shows an example of a drop-down list. When the user wishes to make a selection from the list, the user clicks on the arrow associated with the drop-down box, the choices are displayed, and the user selects the appropriate item.

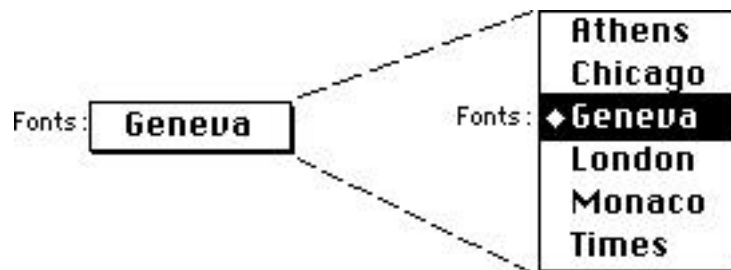


Figure 4-2. A Pop-up menu used in Apple.



Figure 4-3. An example of a drop-down list that allows the rapid setting of styles in Microsoft Excel in Windows.

4.1.1.1.2.1. Proper Usage

⁹ For Motif users, note that the term “pop-up menu” corresponds to options menus rather than to pop-up menus as defined in the OSF/MSG. In Windows, the term “pop-up menu” refers to a floating menu.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- + Use pop-up menus or drop-down lists when
 - the attributes and values on the menu are set infrequently,
 - there are only a limited number of choices available, and
 - screen space is limited.

+ The objects in the group should rarely change.

△ If they change often, consider using a scrolling list instead.

4.1.1.1.2.2. Variations

Pop-up menus in Apple may also contain graphics; more discussion of this variation is found below in Section 4.2.4.

4.1.1.1.2.3. Misusage

√ Only use pop-up menus or drop-down lists to set related attributes and values; they must not contain commands. This is more restrictive than the guidelines set forth by Motif, CUA, and Windows.

√ Do not use pop-up menus when there are over 15 potential choices; use a scrolling list instead (see Section 4.1.1.1.3).

More reading

Apple HIG, p. 88-90.
OSF/MSG, § 4.2.3.
Windows ADG, § 6.3.1.3.
Kobara, p. 112-117.
OL/ASG, p. 143-145, 278-280.
IBM SAA CUA AIDR, p. 184-185.

4.1.1.1.3. List

Lists provide users with a group of attributes or objects (i.e., files) from which to pick. The user sees the list, finds the object of interest (most lists have scroll bars), and then clicks on the object to select it. Once the object or attribute is selected, the application may enable other commands and/or attribute choices.

4.1.1.1.3.1. Proper Usage

- + Use lists for setting attributes or picking objects when
 - items are not selected frequently,
 - the set of items is large, and
 - choices can change frequently.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Figure 4-4 shows a list from which the user can select a single item. Note that the scroll bars should appear (although disabled) even if the list cannot scroll.

Search For:



Figure 4-4. Use a list to reflect choices when those items vary as the application runs or when users are able to edit the items as they appear on the list.

+ Use lists on dialog boxes where the user must select a file or a predefined style. Motif provides some compound widgets, whose principal component is a list for handling objects: `commanddialog`, `selectiondialog`, and `fileselectiondialog`. These widgets allow the user to select one object from among the many that are available. Do not build other kinds of list objects to handle these transactions when the standard widgets will work.

4.1.1.1.3.2. Variations

One variation on the list is the spin button (Windows calls it a “spin box”; IBM CUA SAA AIDR, p. 246, Windows ADG, § 6.4.2.). The spin button has a text box and two arrows that allow a user to display in sequence a list of mutually exclusive choices. For example, a spin button could be used to display the months of the year.

+ Spin buttons should only be used for highly predictable discrete sets of values (e.g., months or days of the week).

4.1.1.1.3.3. Misusage

The list box is flexible enough to handle most kinds of mutually exclusive choices. However, list boxes require more work on the part of the user than do other kinds of mutually exclusive controls. Thus, even though a list box *can* be used, it does not mean that it is ideally suited for the job. Consequently, do not use a list box if another control is more appropriate. Refer to Table 4-1 to select a more appropriate form.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

OSF/MSG, § 4.1.2.1.
Windows § 6.3.
Kobara, p. 134-136, 153-154.
OL/ASG, p. 216-221.
IBM SAA CUA AIDR, p. 49-50, 82-85, 132-133.

4.1.1.1.4. Radio buttons (or as Windows calls them Option buttons)

Radio buttons are probably the most common way to present mutually exclusive choices to the user. The user makes a selection by clicking the button of one of the items in the radio button group; selecting one item deselects the previously selected item.

4.1.1.1.4.1. Proper Usage

√ The labels on radio buttons must not change.

+ Use radio buttons for setting attributes or values when

- there are eight or fewer options,
- there is sufficient screen space, and
- the user does not need to change settings frequently.

+ Put a group box around the radio button group for visual separation, such as indicated in Figure 4-5.

Δ If available screen space is limited, consider using a pop-up menu or list box.

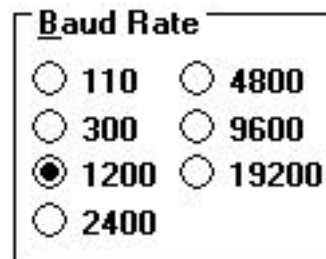


Figure 4-5. Radio buttons provide a clear and efficient way for users to set attributes or values, especially when discrete and few in number.

4.1.1.1.4.2. Misusage

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Radio buttons must not to be used for commands; however, selecting a radio button can be used to activate other controls.

√ Radio buttons convey an exclusive-or relation among a *group* of items and must not be used by themselves; that is, there should always be more than one radio button in a group.

+ When arranging a set of related radio buttons, make sure that the choices are labeled clearly and kept visually cohesive. Figure 4-6 shows an example where the designer used a set of radio buttons, it is not clear which button controls which frequency setting. Generally, it is easier for the eye to scan up and down a list of radio buttons than to scan left to right. While this particular case is not dramatic nor particularly error prone, with some thought the designer could have done better.


poor practice: 

Figure 4-6. An example of the use of radio buttons that might have been laid out more carefully.

More reading

Apple HIG, p. 57.
OSF/MSG, p. 7-96.
Windows ADG, § 6.1.2.
Kobara, p. 67-72.
IBM SAA CUA AIDR, p. 201-202.

4.1.1.1.5. Toggled-menu items

A **toggled-menu item** is a single menu item that can be used to issue two opposite commands (i.e., binary commands). Figure 4-7 shows an example of how a toggled-menu item works. When the user selects `Show Grid`, in a draw or paint application, a background grid would dynamically appear. In addition to the grid appearing on the user's display, the `Show Grid` entry on the menu would toggle to read `Hide Grid`. If `Hide Grid` is then selected, the process would reverse.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES



Figure 4-7. Shows the use of a toggled-menu item. Depending on the current condition (i.e., whether the grid is currently displayed or not) the menu item would change to reflect the command indicating the opposite action.

4.1.1.1.5.1. Proper Usage

+ Use toggled-menu items for two and only two *opposite* commands that are accessed frequently (e.g., `HIDE RULER/SHOW RULER`, `TURN X ON/TURN X OFF`, etc.).

+ Since toggled-menu items are commands, they should begin with verbs (not adjectives!) that unambiguously represent the outcome of the command.

4.1.1.1.5.2. Variations

One alternative to toggled-menu items is to put two controls on a menu. For instance, in Figure 4-7 `Hide Grid` and `Show Grid` could be represented as two separate menu items under the `View` menu title, with dis/enabling doing the work of the toggling. This two-item approach is discouraged because it (1) requires two menu items where one will do, and (2) increases visual clutter. Although if the two commands are not exactly opposite, the two item approach might be best for the sake of clarity (see *Windows ADG*, p. 83).

4.1.1.1.5.3. Misusage

The difference between a can sometimes be confusing. Some commands have the effect of setting attributes. For instance, in Figure 4-8 the `Align` menu is used to align application objects to a grid or to other application objects (e.g., circles in a draw program). Because the `Align` setting is `Left` each new or moved object is aligned to the nearest left grid mark. It might be tempting to implement this as a toggled-command, as demonstrated in Figure 4-7; however, it would be difficult to use a toggled-menu to set the attributes left and right or top and bottom. In toggled-menu items, often the leading command verbs change (e.g., `Hide` and `Show`). When setting attributes, the command

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

itself (e.g., Align) does not change, but the adjective (e.g., Left vs. Right) does. Thus, an attribute menu item is more appropriate for setting these command-like attributes.

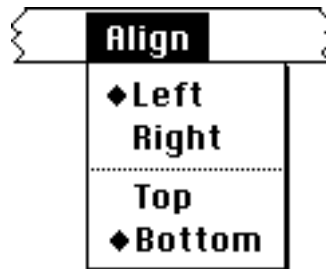


Figure 4-8. A toggled menu item is used to set one item in control. In this case, in both the left/right and top/bottom dimensions the user can set one, and only one, attribute to be in effect in each dimension for alignment.

More reading

Apple HIG, p. 68.
OL/ASG, p. 293-294.
Windows ADG, p. 83.

4.1.1.1.6. Toggled Command Buttons

A **toggled command button** is a single button that can be used to issue two opposite commands (i.e., binary commands). Toggled command buttons are similar to toggled-menu items except that they apply on a dialog box.

4.1.1.1.6.1. Proper Usage

+ Use a toggled command button when the user needs to issue one of two opposite commands while setting options or picking objects on a list. For instance, the user picks an object on a list and depending on the state of the object, the command button will perform the appropriate action. In one state, the button label indicates an action, such as ADD in Figure 4-9. As with toggled-menu items, the labels should be verbs not adjectives.



Figure 4-9. Shows the use of a toggled-command button. In this particular instance, the user picks an item from a list, and if the item is

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

currently on the menu bar the `Delete` label is used. If the item is not on the menu the user can add it by selecting the item and pressing `Add`.

4.1.1.1.7. Icons, icon lists, and icon palettes

Icon lists are sets of icons that are placed in lists much like text lists discussed above, from which the user can choose a single icon. **Icon palettes** are sets of icons fixed in a window (often called “**floating palettes**” or “**toolboxes**”) that convey actions or attributes. Floating palettes are usually available through the application’s menus and can be “torn off”; that is, if the pointer is moved off of the menu with the button pressed, a frame of the menu follows the pointer and is placed at the point where the user releases the button. Icon lists and icon palettes are an effective means of providing options to the user.

Note: There is another class of icons that has properties in and of itself. For instance, a telephone icon that shows its on-hook state with the handset being on the switchhook might change to showing the handset off the switchhook if the user is on a call. These kind of icons will be covered in a future release of this document.

4.1.1.1.7.1. Proper Usage

- + Use icon palettes under the following conditions:
 - Users must change the attribute frequently. If the paint tools displayed in the left panel of Figure 4-10 were available to the user in a menu or a list, the application would be tedious to use.
 - The icons should unambiguously represent concrete real-world objects. If they do not convey a clear meaning, then consider not using icons at all, or possibly using icons with . The best way to find out whether the icons are clear and unambiguous is to test them with users.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

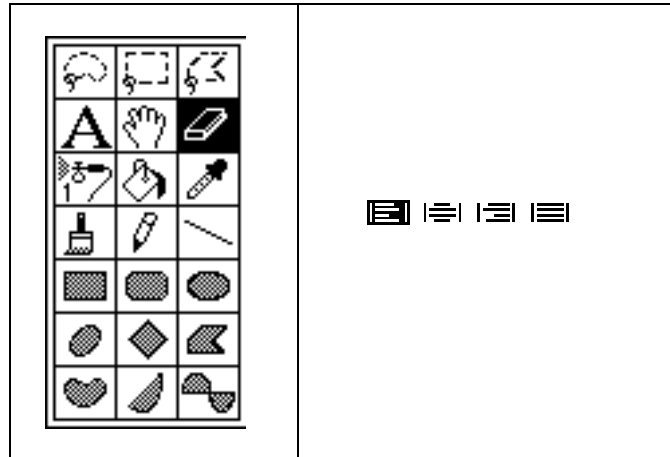


Figure 4-10. Icons can be used to represent attributes and current settings of the user's application environment. The left panel shows a panel of icons used in a paint program. Each of these tools needs to be accessed frequently, and can be reasonably well represented as an icon. In the right panel, the icons signify how the text is to be justified.

4.1.1.1.7.2. Variations

Icons can be presented on fixed palettes, or floating palettes. Use fixed palettes (such the tool sets shown in Figure 4-10) if the attributes on the palette need to be available all the time. Use floating palettes for selections that are needed heavily at certain times and infrequently at others.

4.1.1.1.7.3. Misusage

The use of icons is encouraged where icons accurately and simply convey the meaning of the action or attribute, especially if they represent attributes the user can pick from menus. However, do not use icons when words alone will do the job better than icons (i.e., icons are confusing). In such cases, a pop-up menu (or other mutually exclusive object) would be more appropriate.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 90-92.
OSF/MSG, § 4.1.1, 5.2.3, 5.5.
Windows ADG, § 4.2.8., 7.3.3.
Kobara, p. 181-193.
OL/ASG, p. 294-296.
IBM SAA CUA AIDR, p. 113-114.

4.1.1.1.8. Specialized graphical controls

Specially designed graphics (e.g., a volume knob or sliding bar) can be the most appropriate means of setting an attribute that has a continuous value (e.g., sound level). Since each graphic is unique each would be controlled differently, but the underlying idea is that a graphic should make it easy for the user to interpret what it is and how it works.

4.1.1.1.8.1. Proper Usage

- + Use graphical controls when
 - the set of possible values is continuous with a large range (i.e., the saturation level of the color red), and
 - the attribute lends itself well to graphical depiction or manipulation.

△ When designing these graphics keep the following two things in mind:

- The graphic must unambiguously represent a concrete real-world object. A specialized graphical control should not be used if it is more complicated than another interface object in the toolkit. The only way to find out whether it is harder to use than a toolkit object is to test it with real users.
- Keep the graphics simple. Complicated graphics stand out from the application environment, and probably will not be easily recognizable or usable.

Examples of graphical controls are color wheels on the Macintosh and color sliders in Motif. Figure 4-11 provides an example of the use of a slider to allow the user to manipulate the size of an object through use of the slider. Figure 4-11 also shows the good practice of putting the precise value (i.e., 50%), in addition to the level of the slider, next to the control itself. In fact, put the value in an editable field, so that if the user wants to enter a precise value the user does not have to use the slider.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

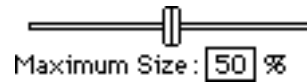


Figure 4-11. An example of a graphical object (i.e., a slider) that conveys a continuous state. This object is familiar and easily interpreted by most users.

4.1.1.1.8.2. Misusage

+ Specially designed graphics can be a powerful way to present mutually exclusive choices, provided that the use of graphics is consistent throughout the application. Unless there is a compelling reason to create a special graphic, use the objects provided in the respective toolkits. Remember that for any specialized graphic, the user will have to spend time figuring out how to use it, when he or she may already know how to control, say, a radio button.

More reading

Apple HIG, p. 57.
OSF/MSG, p. 7-103.
Windows ADG, § 6.6
Kobara, p. 94-98.
OL/ASG, p. 255-259.
IBM SAA CUA AIDR, p. 242-243.

4.1.1.2. Incorrect Forms

There are several incorrect ways of presenting "one of many" choices. Some of them were covered in the preceding section. Listed below is one of the more common errors.

4.1.1.2.1. Check boxes

Check boxes should be used in groups for non-mutually exclusive choices. Often they are used for mutually exclusive binary choices, *but this practice is discouraged*. A check box, in isolation, is not as easily interpreted as a pair of radio buttons, especially to novice or casual users (Figure 4-12 shows an example). A check box requires the user to interpret the meaning of both states of "Network" rather than have them stated explicitly (as in the case of the radio buttons).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

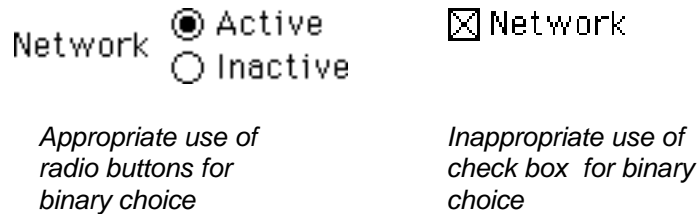


Figure 4-12. Experienced users might not have trouble using or understanding the radio buttons in Panel A or the check box in Panel B. The novice or casual user might find ambiguity of the check box makes it more difficult to interpret than radio buttons.

Figure 4-13 further demonstrates why a check box can be difficult; the wording is often ambiguous. When the box in Figure 4-13 is not checked, it is not clear whether the icon is in the foreground or does not even exist.

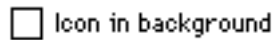


Figure 4-13. Check boxes can often be worded ambiguously when used in isolation to indicate a binary state.

Check boxes may be used if another control is not appropriate and if check boxes unambiguously represent the choice; take care to word check boxes so that the meaning is clear in both the selected and unselected state.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.1.2. “N of many” Choices

When items are **non-mutually exclusive**, (or n of many choices) the user can pick several options from within a group, and all of the items of that group can be active or apply simultaneously to an object under focus. For example,

- a string of text can be underlined, italicized, emboldened, etc.
- a rectangle can have color, a frame width, and a fill pattern in a draw application.
- a printer may substitute fonts, smooth text and graphics.
- multiple files can be selected for processing.

The choice of the recommended interface form depends on the

- (1) number of times that an option is selected,
- (2) number of items in the group,
- (3) space available on the display, and
- (4) frequency with which items themselves can be added or deleted from the list of available choices.

The developer should put most weight on the frequency of selection, i.e., item (1) above. If the user is frequently selecting from the attribute group, make the selection as easy and available as possible; in this case, a fixed icon palette or a pull-down menu with attribute menu items would be appropriate.

Table 4-2 presents the selection criteria in the left column and the recommended interface form in the right column. If your criteria do not fit exactly into one of the boxes on the left column choose the one that is closest to the kind of items you have. Each of the interface forms and its variants is discussed in detail below.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

If Non-Mutually Exclusive Items are...	Then Use the Following Form...
<p><i>attributes</i> or <i>values</i> that</p> <ul style="list-style-type: none">• are selected frequently• are few in number (2-8 choices)• are best represented verbally, and• change rarely.	Attribute menu items
<p><i>attributes</i> or <i>values</i> or <i>objects</i> (i.e., file names) that</p> <ul style="list-style-type: none">• are selected infrequently• are potentially large in number• are best represented verbally, and• can change frequently.	Multiple-selection lists
<p><i>attributes</i> or <i>objects</i> or <i>values</i> that</p> <ul style="list-style-type: none">• are selected infrequently• are relatively few in number (8 or fewer)• have sufficient screen space available, and• do not change.	Check boxes

Table 4-2. This table presents the conditions for selecting the proper interface form for non-mutually exclusive choices.

4.1.2.1. Interface forms

4.1.2.1.1. Attribute menu items

Attribute menu items located on a pull-down menu provide the user with a set of items from which the user can make several selections. In other words, these are non-mutually exclusive choices. The items themselves are attributes rather than actions or commands. When selected, these attributes remain in force until they are deselected by selecting them a second time.

4.1.2.1.1.1. Proper Usage

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ Use an attribute menu for displaying non-mutually exclusive choice groups when:

- options need to be frequently accessed,
- the number of options in the group is small (i.e., no more than 8), and
- when the choices themselves change infrequently, if at all.

Figure 4-14 shows an example of an attribute menu. The Style menu is used to pick which style is to be used at the insertion point (or applied to a group of selected text) in a text editor. In this case, **Bold** and Underline have check marks next to them, meaning the current text will be both emboldened and underlined. If the user next picked *Italics* a checkmark would appear next to *Italics* and the current text would be emboldened, underlined, and italicized.

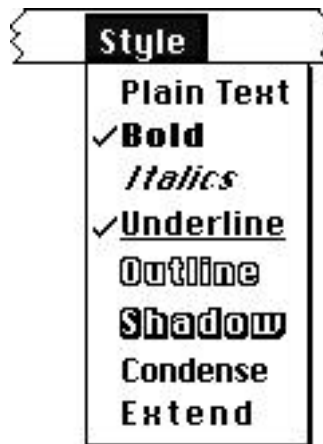


Figure 4-14. An attribute menu can be used to set several items in force. In this case, the **Bold** and Underlined items are selected.

Note: According to strict standards, there should not be interdependencies among the settings in a non-mutually exclusive choice group. However, in the case of Figure 4-14 the text cannot be both **c o n d e n s e d** and extended at the same time, so selecting one has to deselect the other. While this is technically a violation of the standard, in practice “non-mutually exclusive” means that not all combinations are possible.

△ Include one option that deselects all other options (e.g., *Plain text* in Figure 4-14). The rationale is if the user wants to globally disable all the settings, it makes sense to allow the user to do so with one simple action. In situations where this logic makes sense, you are encouraged to employ such an option and create dependencies.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.1.2.1.1.2. Variations

Non-mutually exclusive attribute menu items can be implemented on cascaded menus, if menu space is a limiting factor. See Section 4.1.1 for more details.

4.1.2.1.1.3. Misusage

√ Attribute menu items must not be used as commands.

More reading

Apple HIG, p. 68.
OSF/MSG, p. 7-15.
Windows ADG, § 5.1.1., 5.1.3., 5.2.2.1, 5.3.1.1.3.
Kobara, p. 64-66.

4.1.2.1.2. *Multiple-selection lists*

Multiple-selection lists present the user with a set of attributes or objects (e.g., files) from which to pick. Once the items are selected, the application gives the user a variety of command and/or attribute choices which to apply to the items on the list.

4.1.2.1.2.1. Proper Usage

+ Use a multiple-selection list box for non-mutually exclusive choices when:

- there are a large number of items available to the user, and
- the group needs to be accessed infrequently; from a user standpoint, list boxes can be cumbersome - use them sparingly.

+ Lists are also a good choice if the user can edit the labels of items that show up in the scrolling lists, or add and delete items themselves. For example, in file dialog boxes the list contains the names of all the files available for opening, and the user ultimately has control over the names of the files. Presenting files in anything other than a list would require changing the label of a widget every time it appears.

Most of the time list boxes are used for mutually exclusive selections involving objects, like fonts, files, data records, etc. In Figure 4-15 the user is picking several different (non-contiguous) items to install into the application. While it is possible to use list boxes for selecting attributes, it would be awkward to use a list box to select multiple-text attributes like underlined, italicized, etc.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES



Figure 4-15. Use a list to present choices when those items vary as the application runs or when users are able to edit the labels of items as they appear on the list.

4.1.2.1.2.2. Misusage

Note that a list is used for picking items only. It is not used for showing the user the status of a group of settings. Check boxes and attribute menu items not only allow the setting but also, by their design, they allow the user to see their status. List boxes do not do this.

A list box *can* be used for most non-mutually exclusive settings, but that does not mean that it is ideally suited for the job (see previous paragraph). Do not use a list box if check boxes or an attribute menu is more appropriate. List boxes simply require more work on the part of users than do other kinds of non-mutually exclusive controls.

More reading

Apple HIG, p. 107.
OSF/MSG, § 4.1.2.2; p. 7-59.
Windows ADG, § 6.3.2.
Kobara, p. 56-57, 142, 216-221.
OL/ASG, p. 134-141.
IBM SAA CUA AIDR, p. 132-133.

4.1.2.1.3. Check boxes

Check boxes are commonly associated with non-mutually exclusive choices; they allow the user to simultaneously choose from among a group of items. The operation of check boxes is simple: when the check is in the box the attribute is active - when there is no check, the attribute is not active. In Figure 4-16, the button property for `Bold` and `Shadow` are active, the rest are inactive.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

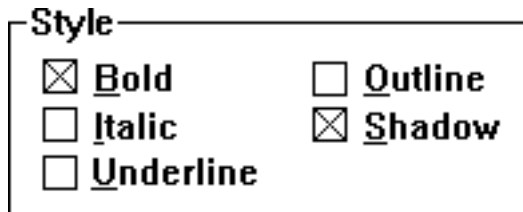


Figure 4-16. Use check boxes to present the various attributes of objects that the user can change. This particular instance allows the user to set various properties of text.

4.1.2.1.3.1. Proper Usage

- + Use check boxes on dialog boxes when
 - the user changes attributes infrequently, and
 - there is sufficient screen space.
- + Check boxes should be used in groups of two or more.

4.1.2.1.3.2. Misusage

√ Check boxes must not to be used for commands; however, selecting a check box can be used to activate other controls.

For small groups of items (eight or less) a set of check boxes is the correct form. For larger groups, you should consider another interface form¹⁰ or, most probably, think about simplifying the task.

- + Check boxes should *not* be used in isolation to indicate a mutually exclusive binary choice; see Section 4.1.1.2 for more information.

More reading

Apple HIG, p. 56.
OSF/MSG, p. 7-15.
Windows ADG, § 6.2.
Kobara, p. 64-66.
OL/ASG, p. 213-214.
IBM SAA CUA AIDR, p. 37-38.

4.1.2.2. Incorrect Forms

¹⁰ A list should probably not be used. The problem with using a list as an alternate for check boxes is that, generally, lists are best used for presenting objects, not attributes of objects.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

There are several incorrect ways of presenting *N* of many choices. Some of them were covered in the preceding section. One of the more common errors is using radio buttons as if they were check boxes. Radio buttons are meant to be used for mutually exclusive or interdependent choices; check boxes are for non-mutually exclusive choices (see Section 4.1.1.1.4).

There might be some confusion over whether to use a pair of radio buttons or a single check box to indicate the setting of a binary attribute, such as Figure 4-12. Lone binary attributes should be implemented as radio buttons, and not as check boxes (discussed in Section 4.1.1.2).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.2. Issuing commands

Commands are actions that modify the properties of objects or manipulate objects in ways described by the command label. When the command is selected, it executes the action immediately. For instance, when the user selects `Rotate object` the application should rotate the object on its axis. Menu items and buttons are the two interface objects (controls) that present commands to users; they have a different look, but each is appropriate under certain conditions.

The choice of the correct interface form depends on the:

- (1) number of total commands the application has,
- (2) frequency with which commands are selected, and
- (3) whether the command is used in association with another control.

√ Do not change the function, interaction style, or appearance of any commands that are part of the standard tool set. For example, do not change the `Copy` menu item in the `Edit` menu to mean duplicate a selected object. Use the predefined commands and their functions.

√ Accompany every menu and button selection with the appropriate feedback for that control (see Section 5.3.5).

+ Use of pop-up menus for commands is discouraged. Pop-up menus provide no clue to their existence; they are visible only when selected by the user. Only consider using pop-up menus if there is very little space on the client area for a panel of buttons.

Table 4-3 presents the selection criteria in the left column and the recommended interface form in the right column. Each of the interface forms and its variants is discussed in detail below.¹¹

¹¹ Icons can be used on buttons or, less commonly, on menus. Since the conditions of their use are not different from the use of buttons, they are not included as a separate entity in Table 4-3. Icons are covered in a Section 4.3.4. If you are considering using icons as a means of providing commands, you should review both the sections on buttons (Section 4.3.2) and the section on icons (Section 4.3.4).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

If You Have...	Then Implement Commands as...
<ul style="list-style-type: none">• Six or fewer total commands• A simple application, and• An application that does not require standard menus.	Buttons in the window
<ul style="list-style-type: none">• More than six total commands (not counting standard menus) that can be arranged into groups.	Menu items on a pull-down menu
<ul style="list-style-type: none">• commands used with other controls• A complicated command requiring additional information to complete.	Buttons in dialog box
<ul style="list-style-type: none">• commands that are selected frequently affecting the entire window; that is, when selecting a menu item is so frequent that it is tedious to use the menu bar.	Buttons in window or a Ribbon
<ul style="list-style-type: none">• Occasional heavy use of commands.	Floating palettes

Table 4-3. Selection criteria for picking the right interface object for commands.

Before discussing the specific interface types, some general issues surrounding labeling and navigation common to both buttons and menu items will be discussed.

4.2.1. Labeling Commands

√ Label each command.

√ When commands are unavailable, signal their status by dimming (also known as graying or disabling) the menu item or button.

√ Use an ellipsis (...) to indicate that a dialog box follows the selection of the command.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Exception: Do not use an ellipsis when the following dialog box is a confirmation or warning box (see below).

√ When deciding on names for application-specific menu items and mnemonics, you must first assign all system-defined menu items and their mnemonics.

- + The following are recommended practices in labeling commands:
 - Use labels that indicate the purpose of the command or the result of what happens when the command is selected;
 - Use short, clear, concise words familiar to the user; do *not* use sentences as labels;
 - Use the active voice;
 - Make menu titles distinctive from each other;
 - Capitalize the first letter of the menu item. With menu items of more than one word, capitalize the first letter of each important word, as well as acronyms and abbreviations that are normally capitalized.

+ In certain contexts, the name of the command can be changed to clarify the meaning of the command. For example, after a `Cut` operation change the `Undo` command to be `Undo cut`.

4.2.2. Navigation and selection

+ Allow the user several methods for selecting commands, e.g., a pointing device, keys, and keyboard access to menus and buttons through mnemonics.

4.2.2.1. Accelerators and mnemonics

√ When using keyboard selection of menu items or command buttons, make the options selectable in either upper or lower case. For example, `ALT-K` would be equivalent to `ALT-k`.

√ Do not reassign the standard key assignments found in the user interface style guide. Do not assign system-defined keyboard equivalents to any other commands (e.g., do not use `COMMAND-C` to mean communicate). Section 3.4.2 shows standard key assignments in Apple, Motif, and Windows.

+ Applications can use `ALT` (or `COMMAND` and `OPTION` in Apple), `SHIFT`, and `CTRL` key combinations for accelerators. However, the `CTRL` and

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

SHIFT keys must be used in combination with each other; they should not be used individually for accelerators.

+ In assigning mnemonics, make the letters used for keyboard equivalents meaningful. They should be either the first letter of the command or the first letter of an important word in the command.

4.2.2.2. Disabling

√ When a command is unavailable, disable it rather than removing the button or menu item entirely. Task-critical commands should always be available to the user, e.g., `Undo`. If the user attempts to pick an item that is unavailable post a note in the information area explaining that help will indicate why the choice is not available.

4.2.2.3. Target area for cursor

+ Make the selectable area of an option large enough to allow users to point at it without difficulty. Generally, the target area should be the greater of two times the size of the cursor itself (e.g., the arrow head) or six millimeters square.

+ On touchscreens, make the touch area equal to the size of the button, plus a half character around the outside of the label, or no less than 30 millimeters square.

√ Separate touchscreen target areas by at least four millimeters.

4.2.3. Buttons

Buttons execute commands that initiate actions and change objects or alter the interface.

+ Every dialog should have at least two buttons: one closes the dialog and carries out an action and the other closes the dialog box without carrying out any action.

4.2.3.1.

+ Do not use a group box around a set of buttons. Instead use placement, whitespace, and subtle lines to show groups of buttons.

+ The number of buttons on a dialog box should be kept to six or fewer.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ If there is a contingent relationship between a button and other controls, place the button above or to the left of the related controls.

4.2.3.2. Labeling issues specific to buttons

√ Buttons should be titled in the regular system font.

√ Be sure that the label is centered within the borders and that at least two pixels are left between the and the button border.

4.2.3.3. Default buttons for dialog boxes

4.2.3.3.1. Marking the default button

+ A default indication should signify the default button. The default indication is usually a heavy border around the button. Most dialog boxes will have a default button.

4.2.3.3.2. Selection of default buttons

+ Choose a default button that is non-destructive. If the user accidentally presses RETURN, it should not cause undesirable consequences (see Windows ADG, 7.3.2).

+ The default should be a positive response to the dialog. *Exception:* When the results of an action are destructive (e.g., "Are you sure you want to delete that file?"), make the user explicitly select the destructive action.

+ If none of the buttons is destructive, per se, set the default as the button most frequently selected by users.

4.2.3.4. Placement, location, and size

4.2.3.4.1.

+ Place standard dialog box command buttons (e.g., OK) in a horizontal row at the bottom of the dialog box, separated from the rest of the dialog box by a separator (e.g., a subtle line). Buttons may be placed vertically at the right side of the dialog box (see Windows ADG, 7.3.5).

+ If a button is associated with another interface object, put the button near the object it affects.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ If a window can be scrolled, do not scroll the buttons.

+ Do not put buttons in more than one row or one column. Large sets of buttons bombard the user with a large set of choices.

4.2.3.4.2. Size

+ Button size is a function not only of the label, but also of the recommendations by the GUI platform and the importance of the option.

+ Buttons arranged horizontally should all be the same height; buttons arranged vertically should all be the same width.

4.2.3.5. Navigation

+ For command buttons that are frequently selected, assign a mnemonic so that users can select them from the keyboard. But do not assign mnemonics to buttons that have key assignments associated with them already (e.g., ESC for Cancel).

More reading

Apple HIG, p. 10, 56-62, 76
OSF/MSG, p. 7-90.
Windows ADG, § 6.1, 7.3
Kobara, p. 55-61.
OL/ASG, p. 49-50, 210-211, 321-342.
IBM SAA CUA AIDR, p. 195-197.

4.2.4. Menus

Menus allow the user to issue commands and set attributes. See Section 3.4.2 for more information on standard menus.

√ Each menu should have at least two choices.

+ Each menu should contain ten or fewer menu items (i.e., commands).

4.2.4.1. Heuristics for constructing menus

+Use the following heuristics for constructing menus:

- list all the commands to be implemented through menus; be certain to include all required menu items from Section 3.4.2.
- identify groups of related items, and then identify groups of item groups.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Group related items on menus into conventional or natural categories known to users. *Rationale:* The more controls a menu contains, the longer it takes users to pick the desired command; grouping reduces selection time and errors.

Group like menu items. For instance, commands should be grouped together; mutually exclusive menu items should be grouped together, etc. Use separators to visually set off the groups.

- Identify menu titles.
- Sort identified groups into menu categories.
- Order groups by frequency of use, importance, and logical relation, etc.

+ Place important options first in the group (e.g., *Save* goes before *Quit*)

+ Organize selections by *order* of use; for example, place *Copy* before *Paste*.

+ If users are accustomed to a particular option sequence, then use that sequence, e.g., a fiscal year beginning with *July*.

+ Put parallel functions, such as *Import/Export*, adjacent to one another with order of use determining which is first.

+ Separate potentially destructive commands from frequently chosen commands; this helps avoid accidental .

+ When there are eight or fewer items for a group of menu items and the frequency of use of those items can be estimated, put the most frequently used options first; otherwise use an alphabetic order.

4.2.4.2. Labeling

4.2.4.2.1. Title

√ Menu titles must be only single words.

+ The menu title should reflect the kinds of commands (i.e., menu items) present in the menu group.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.2.4.2.2. Menu items

+ Commands should be worded so as to represent actions and commands to the user, rather than questions to the user.

+ Because `Undo` acts with the clipboard and other “hidden” objects, the wording of `Undo` should be changed to reflect what will actually be undone. For instance, before a `Cut` operation `Undo` might be shown simply as `Undo`, but after a `Cut` the label should change to reflect what there is to `Undo`, i.e., `Undo Cut`.

4.2.4.3. Availability

√ Unavailable items do not disappear, they are made unavailable or disabled by dimming or graying.

√ During modeless operation, users must always be able to pull down a menu and see the names of options, even if every option is unavailable at that time.

√ Do not add or remove choices from a menu unless the user takes explicit action to add or remove an item or the user toggles between short and full menus.

More reading

Apple HIG, p. 3-5, 23-27, 58, 65-92
OSF/MSG, § 4.2.3; p. 7-69.
Windows ADG, § 5.
Kobara, p. 107-111.
OL/ASG, p. 282-296.
IBM SAA CUA AIDR, p. 193-194.

4.2.5. Icons

Icons are used to represent commands or attributes. The icon graphic should show a concrete object and clearly represent the action executed or attribute selected if the icon is selected. It should also be easily distinguishable from other icons and their actions. Icons have both graphic and label areas. They are most often associated with use on buttons, but they can be used in menus as well. Although icon use in menus is not common, it can be an excellent means of conveying commands if well-represented graphically (e.g., `Flip Horizontal`).

4.2.5.1. Usage

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ Use icons when they represent well-known symbols or widely used industry conventions and their meaning is clear.

+ Icons are recommended when users have different linguistic backgrounds (e.g., at an international airport), or when user's reading skills are questionable.

4.2.5.2. Problems with icons

+ Complex icons are hard to interpret; keep icons simple.

△ Although icons can make an application *appear* more usable, using a lot of icons is discouraged. Too many icons can make an application appear obscure and unapproachable.

4.2.5.3. Labels and form

+ It is strongly recommended that icons used on have

+ Label icons which look identical, but represent different instances; for example, a document for a word processing application.

+ Make icons visually distinct from each other and their meaning should be easily recognized.

△ Icons should be at least ten millimeters apart. Provide at least a four-millimeter hot spot outside of the visible area of each icon. The larger the selectable area of the icon, the easier the pointing action and the less risk of error.

More reading

Apple HIG, p. 5, 11, 14, 41, 42, 61, 66, 69

OSF/MSG, § 5.5; p. 7-52.

Windows ADG, § 4.2.8.

Kobara, p. 181-193.

OL/ASG, p. 13-17, 237-241.

IBM SAA CUA AIDR, p. 113-114.

¹² The downside of labels with icons is that they add to the visual clutter. But the tradeoff between clarity and should be decided in favor of clarity.

4.3. Getting field input

This section discusses the components of a text entry field, good design practices for entry fields, and then the larger issue of how to organize displays with several entry fields, such as fill-in forms.

4.3.1. Elements of entry fields

An **entry field** contains a label, text box, and sometimes a format aid (see Figure 4-17). Discussion of the purpose, placement and properties of each of these elements follows.

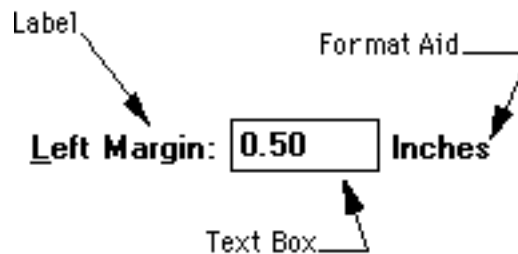


Figure 4-17. Elements of a text entry field.

4.3.1.1.

4.3.1.1.1. Purpose

Labels are short, informative descriptions of the data required in the text box. Labels help users understand the entry that should be made.

√ Always employ the same label to indicate the same kind of data entry. For example, do not use "Zip" one place and "Zip Code" in another.

+ It is *strongly recommended* that *all* text boxes be accompanied by labels. Labels for some text boxes are implied by the context or are so common place within the GUI environment that the individual text box is not labeled.

4.3.1.1.2. Placement

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Labels must be followed by a colon and one space. When the label is placed to the left of the text box, place one space between the label's colon and the text box.

+ When a single label is associated with a single text box, it is *strongly recommended* that labels appear to the left of the text box, rather than above the text box. For example, do this (a)

Date:

rather than this (b)

Date:

Method (a) reduces visual clutter and makes it easier for users to pick out labels from text boxes.

+ Text boxes can appear in either row orientation or column orientation, as shown in Figure 4-18:

- For row orientation, locate the label to the left of the leftmost text box; as with single occurrence fields, separate the label from the text box by a colon and one space. Separate text boxes by at least five spaces.
- For column orientation, place the label above the column of text boxes flush with the left edge of the column of text boxes.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Multiple occurrence fields in row orientation

Key:

Multiple occurrence fields in column orientation

Key

Figure 4-18. Column and row orientation for entry fields.

4.3.1.1.3. Other requirements and recommendations

√ Use mixed case for labels, with initial capitalization on all key words.

√ Make the labels distinctive from text boxes, control options, and messages. Do not place a box around labels that makes labels look like text boxes. As noted above, labels should be differentiated from text boxes by terminating the label with a colon, and by placing labels to the left of the text box.

+ Labels should be easily distinguished from one another; minimal differences (one letter or word) cause confusion.

+ Labels should be words used by the user community, not the words preferred by developers. When in doubt, ask users.

+ Use only standard alphabetic characters; avoid contractions, hyphenations, abbreviations or short forms, except where those abbreviations are in common usage and understood by all users of the application. For instance, "PO" is an acceptable abbreviation for "Purchase Order" because "PO" is widely recognized. Abbreviations and acronyms should *not* include punctuation.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

△ Space allowing, do not stack word in labels, as in the left side of Figure 4-19. *Exception:* Units of measure should be placed below the label (see Section 5.2.1.1.2).

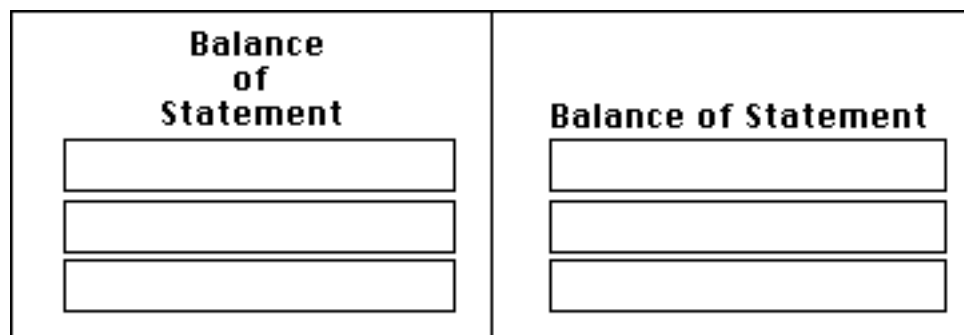


Figure 4-19. Left panel shows stacking labels in a column where there is space to put it all on one line. The right panel shows the preferred way labels should appear.

More reading

OL/ASG, p. 253.
Galitz, § 4-2, 4-3.
Smith, § 1.4.
Tullis, p. 395-397.

4.3.1.2. Text boxes

Text boxes allow users to enter information from the keyboard.

4.3.1.2.1. Placement

+ See Section 4.3.1.1 for information on placement of text boxes. Generally, they are placed to the right of the label.

4.3.1.2.2. Requirements and recommendations

√ Always show the user the text being typed into a text box except if they are entering a password. Even if users are typing a password, provide feedback as to how many characters the user has typed by echoing a non-text character like the “•” in place of the actual character typed.

√ If a window contains required fields, enable the appropriate dialog buttons only when those fields are completed. (The user must always be allowed to cancel.) For instance, an electronic mail message cannot

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

be sent without an addressee. Only after the addressee field has been completed should the SEND button be activated.

√ Do not require users to enter special separator or delimiter characters, such as "-", "\$", etc. This can be done by using the unit of measure in the label or separating multiple text boxes with the proper delimiters.

√ A user must never have to justify an entry to the right or left. If the item length is variable, provide automatic justification when the user leaves the field.

√ Users must never have to count spaces in a text box.

√ Users must be able to enter text in either upper or lower case - users must not be forced into making case-specific text entries for a field unless it is absolutely required by the operating system (e.g., in UNIX).

√ When the user enters numeric data, entry of the decimal point must be optional; report errors through dialog boxes.

√ Allow users to Cut, Copy, Paste, and Undo entries in text boxes.

+ The length of the text box should accommodate about 95% of the entries that the user will type in the box. For instance, the width of a standard street address might be 25 characters. But making the box 25 characters long will not accommodate the 15 percent of entries that are longer than 25 characters. It would be best in this case to expand the width of the text box to 35 characters, where only five percent of the possible entries would be longer than 35 characters.

+ If the user must enter text that should not be echoed (e.g., a password) that has a fixed length, show the length of the field with some non-standard text character. For instance, in Figure 4-20 the text box is initially filled with "•" but as the user types stars "*" fill in to give the user feedback. Do not verbally tell the user how long an entry should be, e.g., "Field is eight characters", this forces the user to count.

Password:

Figure 4-20. Cueing fixed-length fields.

+ Use a constant width font for fixed-length text entries.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ A text box can be either a single line or multiple lines with scroll bars. Single-line text boxes typically scroll horizontally, except when there are application-imposed limits to the length of the entry.

√ Multiple line text boxes must employ word wrapping.

+ Do not force users to truncate field entries, even though they can truncate if they choose. For example, if the application only needs three characters of a field to be entered it is still (usually) faster and less cumbersome to allow users to enter the full word than to force them to truncate after the first three letters of the word.¹³

+ Break up entry and presentation of long numerical and other non-word strings every four or five characters through use of separating lines, dashes or spaces. *Reason:* Users' ability to scan long strings of letters and numbers degrades after five characters. In Figure 4-21, if the code is presented in one long string, as in the left panel, it is much harder for the user to enter and read than if it is separated as in the right panel.

Id Code:

Figure 4-21. The use of separating lines, as shown here, dashes, or spaces should be used to break up the entry of long non-word strings of characters.

+ It is strongly recommended that text be justified left, and numbers be justified right or anchored around a decimal point. Similarly, entry of leading zeros should not be required.

+ Requiring users to recode, change, omit, or add to an data source before keying is *strongly discouraged*. The fewer rules involved in keying, the faster and more accurate data entry will be.

+ Highlight the current field to be entered by using reverse video.

Δ Current or default values should be displayed in text boxes when the dialog is opened; generally, the text in the first or most important text boxes is highlighted.

¹³ Also, when entering state names, allow users to enter the whole state instead of forcing them to enter the two-letter state abbreviations. Firstly, most users do not know all the proper state abbreviations, but they can type the name of the state. Secondly, the computer should be able to make these transformations easily.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

△ Make maximum use of stored data. If the application already has information that is being requested in a field, provide that data in the text box, but allow the user to change it. Do not make the user re-key information that the application already has.

More reading

Apple HIG, p. 106-114, 118-119.
OSF/MSG, p. 7-120.
Kobara, p. 126-130.
OL/ASG, p. 54-55, 214-215.
IBM SAA CUA AIDR, p. 88-89, 257-258.
Smith, § 1.0, 1.3, 1.4.
Lund
Galitz, § 4-2, 4-3.

4.3.1.3. Format aids

Format aids provide an indication of the nature of the desired response. They are primarily used for fields on windows that are infrequently used, or to clarify what is meant by a label. Figure 4-17 below shows a good example of this.

+ In general, use format aids sparingly because they quickly become visual clutter.

+ Place format aids to the right of the text box, rather than between the label and the text box.

4.3.1.3.1. Requirements and recommendations

√ When a measurement unit is consistently associated with a particular text box, include that unit as part of the field label or as a format aid. Figure 4-22 shows an example of how a format aid makes entering the data easier when the unit of measure is uncertain. Furthermore, when the user enters one of the measures the other should be updated automatically. For instance, if the user enters the distance of 15 feet in Figure 4-22, the application should enter "5" in the yards text box.

Distance: **Feet** or **Yards**

Figure 4-22. Use format aids to allow different units of measure.

+ Allow format aids to be turned on/off. For instance, one of the preferences the user might set would be to turn off format aids that were

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

superfluous, like the aid in Figure 4-17. Note that the format aid in the above Figure 4-22 is needed and should not be turned off.

More reading

OL/ASG, p. 253.
Galitz, § 4-2, 4-3.
Smith, § 1.4

4.3.2. Placement of display elements

Applications usually require a set of entry fields to be displayed at the same time on the window. This section presents guidelines on how to design good text entry displays. Figure 4-23 shows a typical data entry dialog box.

Database Query

Client ID: 708-669-0954

Last Name: Smith

First Name: Edward

Company:

Address:

Mail Stop:

City:

State: Zip:

Work Phone:

Find Cancel

Figure 4-23. An example of a data entry form.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.3.2.1. Requirements and recommendations

4.3.2.1.1. *Layout*

+ Guide the user through the window with implicit or explicit lines formed by display elements. Make use of subtle lines to divide the screen and draw the user's attention to the important information.

+ Ensure that the size of the window or dialog box is both long enough and wide enough to support efficient text entry and editing.

Δ Make location on the display compatible with importance. Lay out the display from top left to bottom right; place the more important (and required) fields at the top and to the left. When the window is posted, provide an obvious starting point on the display in the upper left corner by highlighting a field. If the user will be keying from a source document, see Section 4.3.5.3.

Δ Arrange entry fields (and other controls) into logical groupings. There are several common information-grouping techniques. Each is appropriate for certain applications. Some grouping techniques are:

- Relatedness - group entry fields that have logically related information;
- Sequential - group entry fields in the order in which they are commonly received or transmitted;
- Frequency of use - group entry fields that are used most frequently; and
- Importance - group entry fields according to how important items are to the task or transaction.

Δ When the display appears, it should present a clear indication of what information is expected from the user and where that information should be placed. Proper placement of headings, field labels, instructions, and actions/options that the user has available provide this "clear indication." All wording on the display should be in plain, simple English.

4.3.2.1.2. *Amount of information*

+ Provide all data related to one task on a single primary window or in secondary windows. Provide only information that is essential to making a decision or performing an action. Do not flood a person with information.

+ Present information in a directly usable form. Do not require reference to documentation, translations, transpositions, or interpolations.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ Status or error messages should come to the user through message dialog boxes. Non-critical information should be presented to the user in the information area in the lower left corner of the window (see Section 5.3.3.2).

+ Help for entry fields and other user guidance should be provided to the user through use of the help facility. All screen objects should have help available to the users through a moded direct selection (i.e., the user presses the HELP key then points and clicks at an object), the menu bar or as a command button on modal dialog boxes. See Section 5.3 for information on user guidance.

4.3.2.1.3. *Field alignment*

+ Vertically align text boxes and labels into columns as stated above. Left justify text boxes and right justify labels to text boxes as shown in Figure 4-23. *Note:* It is common to see dialog boxes where both the labels and text boxes are left justified. While this is not wrong *per se*, it can place some labels a fair distance from the actual text box. This increases the number of eye movements the user makes, and increases the likelihood of error thus slowing the user down.

+ Leave a minimum of five spaces between the longest text box in one column and the left-most label in an adjacent column (also see Section 5.2.1).

4.3.2.1.4. *Headings*

Sometimes the user needs more guidance than just the labels; headings are needed. Headings are more general labels for entry fields. A record that consists of entry fields for name, street, city, state, and zip could be grouped under the heading "Name and Address".

+ There are several kinds of headings, each is placed slightly differently. Figure 4-24 shows an example of the proper use of headings in a data form.

√ The text-entry group (that is either a dialog box or a frame) must be titled.

- Section headings refer to local groups of related information. Section headings are located in the frame surrounding the text-entry group.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- Sub-section headings refer to sub groups within sections. Locate sub-section headings left of the top-most row of its related fields.
- Indent labels a minimum of five spaces from the start of the sub-section heading.
- Separate the column of labels from the longest heading by a minimum of four spaces.

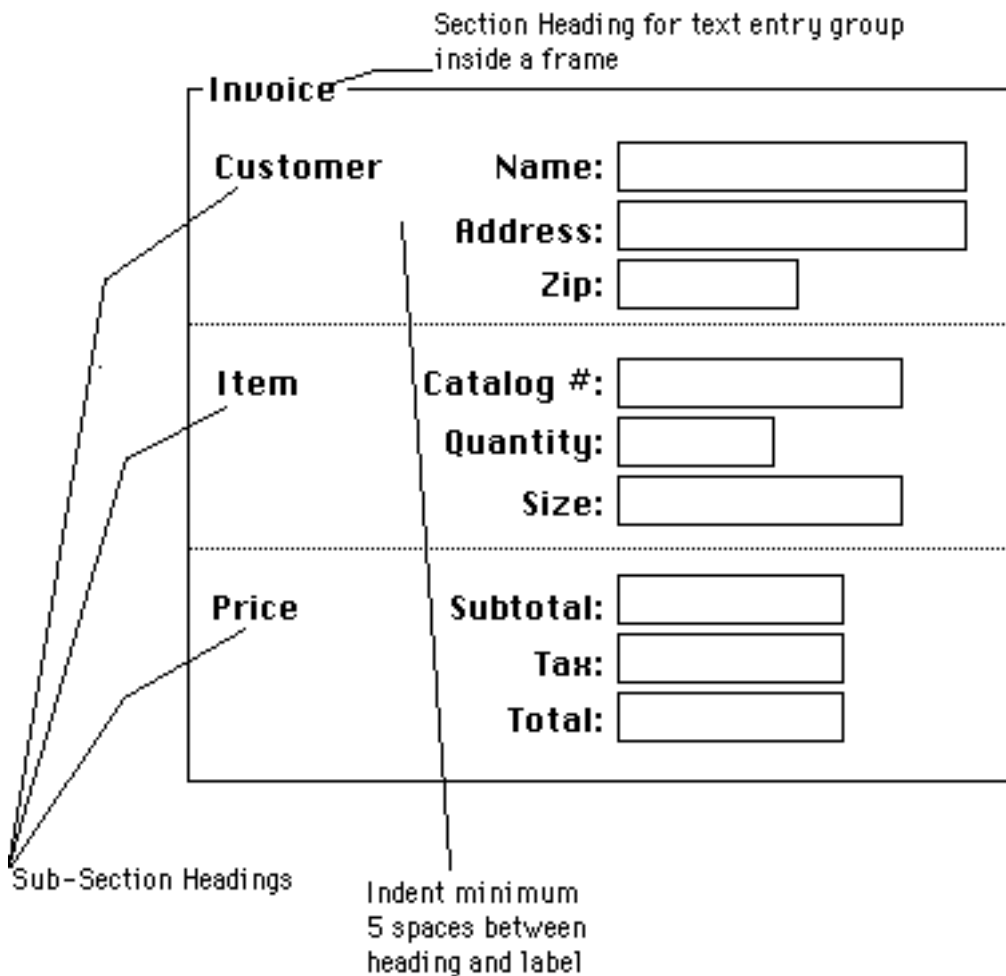


Figure 4-24. Shows the use and placement of headings and sub-headings.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

Apple HIG, p. 58-60.
OSF/MSG, § 4.2.
Windows ADG § 6, 7.1, 7.3
Kobara, p. 195-223.
OL/ASG, p. 263-275.
IBM SAA CUA AIDG, p. 77-81.
IBM SAA CUA AIDR, p. 275-276.
Galitz, § 4-2, 4-3.
Lund
Smith, § 1.0, 1.4.
Tullis, p. 382-406.

4.3.3. Navigation

4.3.3.1. Requirements and recommendations

√ The user must take an explicit action to initiate the processing of entered data; do not initiate processing as a side effect of entering information in the last field of a dialog box.

√ Do not allow users to enter in protected fields; tabbing should take the user to the next field which the user can edit.

+ When a window or dialog box is first displayed, the cursor should be placed automatically in the first text box. That field should be highlighted and ready for user input.

+ It is strongly recommended that you require users to move focus by explicitly tabbing from one text box to the next control in the tab order (obviously, the mouse can also be used to randomly pick a control). The application should *not* move the cursor automatically when the field is complete. The *only* exception to this is when you are using fixed-length fields in a high-volume production environment.

Δ Minimize user actions required for cursor movement from one field to the next; placing required fields before optional fields usually makes data entry more efficient.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

More reading

OSF/MSG, § 2.
IBM SAA CUA AIDR, p. 281-282.
Galitz, § 4-1, 4-2, 4-3.
Lund
Smith, § 1.0, 1.1, 1.3, 1.4.
Tullis, p. 382-406.

4.3.4. Edit checks

Once the user has selected an `OK` or `Done` or similar button on a display, the application performs certain edit checks on the data entered. The following are general guidelines concerning edit checks.

4.3.4.1. Requirements and recommendations

+ Data can be checked automatically after navigating away from a field. Or if two or more fields are related then error checking should be done when all contingencies can be checked. When an error is detected, the application can beep (if it is a simple error) or post a dialog box detailing the error. If possible the error message should be displayed in a modeless dialog box to assist the user in using the content of the message to correct the error. When the dialog box is closed or focus is shifted back to the window containing the error, the field in error should be highlighted.

√ The user's inability to correct an error should not prevent initiation of another transaction (that is, allow `Cancel` if nothing else). If possible, the application should store the last transaction (or set of transactions) that contained an error for its later correction.

+ On-line user guidance or help information should tell the user what edit checks are being applied to a field (see Section 5.3.).

More reading

Windows ADG § 6.10
Galitz, § 4-1.
Smith, § 1.7.

4.3.5. Data forms

Data forms display (view only) sets of entry fields, such as in Figure 4-25.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Database Query

Client ID:

Last Name:

First Name:

Company:

Address:

Mail Stop:

City:

State: Zip:

Phone:

Figure 4-25. An example of a database query form.

4.3.5.1. General

+ Make layout of corresponding data forms consistent across displays. For example, do not ask for first name followed by last name in one display, then ask for last name followed by first name in another display.

√ When the same form is used for data entry as well as for data display, use the same labels and sequence for each.

+ Right justify labels, left justify text boxes.

+ Long data items should be broken into three- and four- character groups separated by blanks or a special symbol. For example, an account number might be "67A5245B74"; it should be displayed to users as "67A 5245 B74". This greatly improves readability.

4.3.5.2.

Labels describe the content of the field. See Section 4.3.1.1 for more information on good practices for labels.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ Each text box must have a label.

√ Identical data must have the same label across displays.

+ All labels must be distinctive; that is, two fields should not have the same label.

4.3.5.3. Use of source documents

√ If paper documents are used for data entry, design forms to be consistent with the layout of the paper source. If the paper source is poorly constructed, consider redesigning the paper form so that there is a good match between the computer form and the paper form.

+ If data entry form does not rely on paper, group fields by function, sequence of use and importance.

More reading

Galitz, § 4-2; p. 165-185.
Smith, § 1.4.25.

4.4. Putting it all together: Organizing controls in windows and dialog boxes

Organizing a display is a very important aspect of designing a good interface. Good organization results from good task analysis and allows the user to comprehend the display quickly and correctly.

Good design:

- enables the user to predict how other displays will look,
- judiciously uses interface objects and screen space,
- is aesthetically pleasing through the use of symmetry,
- has a logical, rhythmic ordering that guides the user's eye through the display, and
- demonstrates simplicity that results in ease in comprehending the interface objects.

The key pieces to organizing the display are the layout of the elements and navigation among the elements.

This section proceeds with a discussion of essential principles and practices of window and dialog box layout (see Section 3.4.3. for standard dialog box types). Following the discussion of principles and practices is a section on navigation within controls on a window or dialog box.

4.4.1. Good practices of display layout

The basic principles of window layout are the same as for dialog box layout.

△ Good display layouts depend on providing the correct amount of information with good organization. Not providing enough information is inefficient; providing too much is confusing. It is essential that during design you examine each control on a window or dialog box and ask whether that object should be there or not and whether other controls might be needed.

△ Make objects and groups of objects visually distinct. A number of factors impact a user's comprehension of the display and efficiency of interaction with the display: total number of items, grouping of items, alignment of groups, length of items, labeling of groups, ordering of

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

items, and density of groups. Figure 4-26 graphically shows examples of these concepts. Each is discussed later in this section.

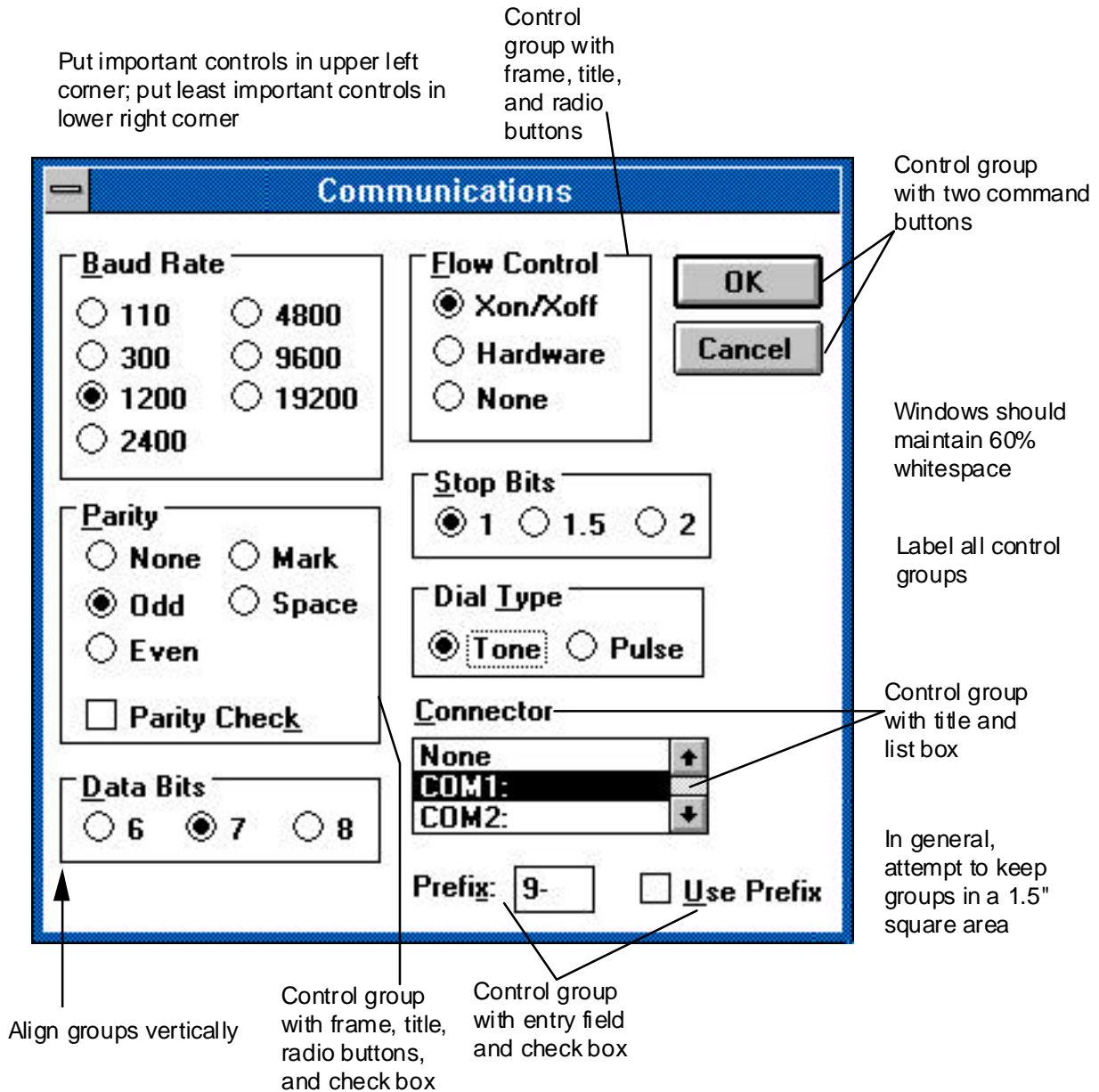


Figure 4-26. Organizing principles and practices of dialog boxes and windows.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

There are some rules of thumb based on human perception that serve to anchor the design layout. Below you will find specific discussion and examples of good practices for the layout of dialog boxes and windows.

4.4.1.1. Grouping controls

Groups are defined as controls that are conceptually related or similarly structured, and surrounded by white space or subtle lines. Grouping aids recall and results in faster processing. Examples of an organized group would be a set of radio buttons, a list box, or a pop-up menu. It is sound practice to surround radio button and check box groups with a labeled box.

Grouping heuristics:

- Function - All information that is processed together or thought of *by users* as being related is grouped together. After the items are grouped, another grouping technique is used for sequencing items in each group and for sequencing groups.
- Frequency of use - Information is grouped in order of the frequency with which it is used, with the most frequent first.
- Sequence of use - Display information in the same order in which it is required by the user's task.
- Location or related format - Information is displayed/requested in the same relative position on every format used in the same application or by the same users.
- Importance - Group information in order of the decreasing importance to the user.
- Habitual Sequence - Group information in the order in which users normally conceive of it.
- Alphabetic, numerical, or chronological ordering - Only if all of the above do not apply, group information in alphabetic, numerical, or chronological order.

+ Place no more than 20 organized groups, and usually far fewer, of information on a single window or dialog box. Figure 4-26 has 10 organized groups.

+ Groups should fit within the user's visual focus. Roughly, at average grouping distances, that means 7 lines high by 14 characters wide.

+ Any particular window or dialog box should have no more than 40% of it filled with words or graphics; it should contain at least 60% white

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

space. **White space** is defined as the number of display locations that do not have any characters or graphics.

+ Spacing between groups and objects in dialog boxes should be more generous than in windows. The reason is that users have to interact with windows more than with dialog boxes. Consequently, there is greater need for efficiency of placement in windows. Dialog boxes, on the other hand, need to be rapidly interpreted which means better use of grouping, spacing and layout. Spacing between buttons should be twice as far in dialog boxes than it is in windows.

4.4.1.2. Locating controls

+ Put the most important and frequently used controls toward the top and left of the window or dialog box.

+ Design the controls and sets of controls so that the flow of interaction through the window or dialog box requires as little cursor travel as possible.

+ Place command buttons that affect the entire window or dialog box horizontally at the bottom, with the group of buttons centered. There are some exceptions to this, see Section 4.2.3.4 on button placement.

+ Do not put buttons in more than two rows. If you find that you have a lot of buttons, consider using a menu or a toolbox or ribbon.

4.4.1.3. Dependencies among controls

+ Locate controls so that controls at the top affect controls below. There should be a logical sequence suggested by the layout of the controls. If one control enables another, the enabling control should be above or to the left of the enabled control.

4.4.1.4. Display size

+ In general, you should not make the default size of a window the full screen.

+ You should not make dialog boxes that are larger than the default size of the primary application window.

+ Make the initial size for each window large enough to accommodate the amount of data that you expect a typical user to see.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

4.4.1.5. Group titles

√ Titles should *not* be underlined. Underlining on a CRT makes reading difficult. This rule applies to almost all text displayed on a CRT. See Section 5.1.5 on highlighting and coding.

√ Groups of command buttons should not be titled. Instead their actions should be obvious from the button titles themselves.

+ Most groups of objects should have titles (see Section 4.3.2.1.4). Make titles reflect the kind of information being sought by the user.

+ Titles should be distinctive from one another.

4.4.1.6. Size of display

√ Maintain column and row headings when paging/scrolling through tables.

+ If one dialog box calls another, make the last called dialog box movable, if at all possible.

+ The application determines what happens to the controls inside a resized box, whether they are truncated, resized or rearranged. In general, it is not a good idea to allow users to resize dialog boxes. You must define a minimum size for the window that maintains essential information. See the specific style guides for how to resize.

+ When the data exceed the available display area, give the user an indication of where the currently displayed data are, like a page number in the information area, relative to the rest of the document.

+ The initial size and layout of a window should be such that a user will not need to scroll the window. If the window is not big enough, put less frequently used controls out of view.

4.4.2. Navigation

√ Move the keyboard focus with the TAB key.

+ All controls should be accessible through the keyboard.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ If the new focus is a single-line text box containing editable text, select all the text in the text box when the focus enters the text box. If no text exists in the box, then place a location cursor in the text box.

+ When a window or dialog box is opened, the cursor should be positioned at the object most likely to change or be completed. If the focus is returned to a window, then the cursor should be placed on the last component in that window to have a location cursor.

+ Cursor movement from object to object and group to group should be from upper-left to the lower-right.

+ Design the controls and sets of controls so that the flow of interaction through the window or dialog box requires as little cursor travel as possible.

+ If there is a default button on a window or dialog box, pressing ENTER or RETURN should result in entry of all items in a dialog box regardless of where the cursor is placed.

Δ Design the application so as to minimize the number of times the user has to travel between the keyboard and the mouse. If the user is constantly going back and forth, it diminishes performance.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5. Guidelines for Providing Output to the User

Contents

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.1. General

5.1.1.

√ Users must *not* be able to type over or change display-only data.

+ Display data in common units according to cultural standards. For instance, miles per hour is the accepted unit for speed in the US, however most of the rest of the world uses kilometers per hour. Make units consistent across the entire application.

+ The displayed data should be directly usable. For example, do not display a field in feet when the user needs yards or meters. If more than one unit is used, display data in all used units (see Section 4.3.1.1).

+ Labels and data should be in words familiar to users.

Δ Maintain consistent formats across displays. Use consistent wording and grammatical structure for data and labels both within and across displays. Titles and headings should not change across multi-page displays of the same table or form.

Δ Ensure that field labels are consistent in word choice, format, and style with the presented information. For instance, if the label for a field is "Last Name, First Name:" the data displayed in the associated field should be presented "Smith, William" not "William Smith."

Δ Display only data relevant to the user's task. Do not place extraneous data on the display.

Δ Locate the most frequently requested information early in the display sequence.

Δ Each display should provide sufficient context to interpret the display. Practically, this means that certain information must be repeated to help the user maintain his orientation.

5.1.2. Avoiding visual clutter

One measure of visual clutter is **screen density**. Screen density is determined by the percentage of character positions on the screen containing data. Typically, screen density measures are used for

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

character-cell terminals. The concept can be extended to the GUI environment by assuming that a "screen" is a "window," and density is measured by the percentage of pixels filled instead of percentage of character positions filled. While screen density is a crude way to evaluate visual clutter it is a useful rule of thumb.

+ Density levels should usually be no higher than 40% (Tullis, 1988).

Δ Items and groups of items should be perceptually distinct. For example, distinct elements should not touch each other. A button's label should not touch the button's border. Explanations of good practices in design can be found in Section 4.4 of this document.

5.1.3. Data groupings

Organize the data in the client area into semantically related groups. Some common are:

- Sequence of use
- Function
- Importance
- Frequency
- Chronological
- Alphabetical

+ "Sequence of use" is the most important grouping technique. Format displays alphabetically only for large sets of data (i.e., more than 50 items) or as a last resort.

+ An optimal size for data groups (not continuous text) is about five degrees of the user's visual angle. Five degrees of visual angle corresponds roughly to 7 lines high, and 14 characters wide (assuming a 10 point constant-width font such as Courier). *Note:* Obviously, it is unreasonable to assume that all data groups could or should meet such a constraint. However, when design permits flexibility, this recommendation should be taken seriously.

+ Use data organizations and formats that users are familiar with. For example, the sequence name, address and phone number is more familiar to users than address, phone number, and name.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.1.4. Abbreviations

+ Use of abbreviations in labels is *not* recommended. *Exception:* Abbreviations are acceptable if they will be understood by all users and/or there are serious space constraints.

+ When abbreviations must be used:

- Ensure that all abbreviations are unique and understood by all users;
- Choose common (familiar) abbreviations, but
- If no abbreviation set exists, design the abbreviation set according to a consistent rule. Vowel deletion (e.g., POLICY becomes PLCY) is usually the best choice for headings and labels. User testing of all the new abbreviations is strongly recommended before they are included in the application. Truncation (e.g., POLICY becomes POL) is usually best for when users have to type in entries.

√ Do not punctuate abbreviations. *Exception:* Punctuation is acceptable for the sake of clarity.

+ If abbreviations are used, you should provide the user with an electronic listing of all the abbreviations in use, perhaps through the help facility or on a floating palette.

5.1.5. Highlighting and coding

5.1.5.1. Purpose and use

Highlighting makes critical or important information perceptually salient to the user. There are a number of ways to highlight information, for example using color, shape, or text attributes. As important as *how* to highlight something is *when* to highlight something.

+ Use highlighting when:

- focusing users' attention on unusual data values
- specifying information that needs to be changed
- calling attention to high priority fields or messages
- indicating next data or command entry areas

Coding refers to the differentiation of levels of a variable or properties of data so that the user can make decisions based upon the coding. For example, in a list box unavailable items should be grayed out or italicized and available items should be in the regular system font.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.1.5.2. Recommendations

+ No more than 10% of the display should be highlighted at any one time.

+ Provide users with a legend explaining the highlighting and coding. A floating palette is one way to provide such a legend.

△ Highlighting and coding are most effective when used sparingly.

Table 5-1 presents a summary of the highlighting and coding recommendations.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Highlighting/ Coding Technique	Pro	Con	Use for
Blinking (Blink rate should be 2-3 hz, and the "on" cycle should be at least 50%.)	Excellent attention-getting capability	Reduces readability Distracting	Highlighting situations requiring urgent attention and a quick response. <i>Note:</i> Be sure to turn off blinking when user responds. Do <i>not</i> blink the whole word or phrase on and off. Instead blink the item in by alternating normal and high brightness. Or use an item (e.g., an asterisk) that is not part of the critical item and blink that item on and off.
High brightness (i.e., boldface) The brighter item should be at least two times brighter than the normal item.	Good attention-getting capability Least disturbing of all highlighting	Can be difficult to discriminate between brightness levels	Highlighting errors or differing screen components <i>Note:</i> No more than two levels of brightness should be used.
Reverse video	Good attention-getting capability	Can reduce legibility Use sparingly	Highlighting errors or important screen objects

Table 5-1. Highlighting and coding techniques (cont'd on next page).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Highlighting/ Coding Technique	Pro	Con	Use for
Color coding	Good for showing relationships among screen elements. Colors have stereotypical meaning to users (see Section 3.3.1.)	Can be difficult to discriminate between colors (see Section 3.3.1)	Coding data <i>Note:</i> Limit the number of colors to four or fewer. <i>Note:</i> Because as many as 8% of users may be color deficient, use color coding redundantly with another coding method, e.g., shape.
All upper case characters	Salient in text display of mixed or lower case fonts	More difficult to read than mixed case	Displaying labels
Mixed upper and lower case characters	Best for readability	None	Displaying headings and labels
Different	Unobtrusive	Can be difficult for the user to notice. Fonts have no stereotypical meaning to users; all associations have to be learned.	Differentiating between application displayed (i.e., unchangeable) text and user-entered text and for coding items in lists and tables <i>Note:</i> Limit the number of different fonts for coding to three.
Italics	Unobtrusive	Has marginal stereotypical meaning to users (i.e., "this is special"); meaning has to be learned.	Coding items in lists and tables

Table 5-1. (cont'd). Highlighting and coding techniques (cont'd on next page).

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Highlighting/ Coding Technique	Pro	Con	Use for
Underlining	None	Poor attention getting Reduces readability	Do not use
Double size characters (must be at least 50% larger than next smallest size)	Good attention getting quality; not visually disturbing	Consume screen space	Displaying titles and headings

Table 5-1. (cont'd). Highlighting and coding techniques.

5.1.6. Labels

A **label** is a descriptive title phrase, or word, consistently placed with a control or group of controls that identifies those controls.

√ All controls must be labeled. Follow guidelines in Section 4 and later in this section when labeling.

√ All labels must be distinctive from each other within and across windows. For instance, "Number" must not mean phone number on one display and social security number on another.

5.1.7. Fonts and font sizes

√ Use the standard system fonts and sizes for labeling all interface objects. Most system fonts are sans serif fonts.

+ Use different font styles to differentiate between text the application displays and text the user controls. In general, *serif* fonts are more readable than *sans serif* fonts, and are thus recommended as user fonts. Times, Roman, or New Century Schoolbook fonts are recommended.

+ In text boxes that require entries of a specific length, use a constant width font (e.g., Courier, Monaco, Geneva); make the text box only

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

accept entries of the specific length. *Note:* This eliminates errors by constraining user input to only those entries that conform to the length requirements.

+ The minimum stroke width of a font should be 2 pixels.

+ Font sizes should never be less than 10 point.

+ Recommended character heights (as measured from the top of an upper case "H" to the bottom) as a function of viewing distance (in inches):

Viewing Distance	MAX	Best	MIN
12	0.09	0.06	0.05
16	0.11	0.08	0.06
20	0.14	0.10	0.08
24	0.17	0.12	0.10
28	0.20	0.14	0.11
32	0.23	0.17	0.13

+ Recommended character widths (as measured from side to side of an upper case "H") as a function of viewing distance (in inches):¹⁴

Viewing Distance	MAX	Best	MIN
12	0.06	0.05	0.04
16	0.09	0.06	0.05
20	0.11	0.08	0.06
24	0.13	0.09	0.07
28	0.15	0.11	0.08
32	0.17	0.12	0.10

+ The recommended space between letters in a word is .004 inches (or roughly 2 pixels).

¹⁴ Since the width of a font is proportional to the height of the font increasing the font height automatically makes the font wider. Thus, a font 9 pixels high will have a width of 6 pixels; a font 8 pixels high will be 5 pixels wide.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ The recommended space between lines (i.e., the baseline of one font and the top of the next line) is .020 inches (or roughly 10 pixels).

More reading

Kobara, p. 33-41.

5.1.8. Searching text or lists

+ Users should be able to search for specified strings in large bodies of text or long lists. Expert users might wish to search on special characters (e.g., carriage returns) or formats (e.g., italicized text).

+ When searching, upper and lower case letters should be treated as equivalent in search unless explicitly noted by the user.

+ Users should be able to globally search and replace one string with another without requiring confirmation for each change. Also allow them to undo *all* changes immediately following the replace.

+ The case of the replace string should be the same as the search string.

5.2. Information displays

5.2.1. Tables

Tables are a critical means by which data are presented to users. Figure 5-1 below shows an example of a table.

Name	Number	Street	City	St	Amount	
Smith, John	555-2323	1010 Washington	Peotone	IL	153.25	↑
Jones, Betty	555-2324	123 Fischer	Pontiac	IL	5.20	
Brown, Barb	555-2325	838 E Broad	Paxton	IL	10.00	
Wells, Will	555-2326	333 Chrisman	Plano	IL	999.75	
Doggs, David	555-2327	451 Jupiter	Paducah	KY	50.00	
Williams, J.	555-2328	1973 Elm	Pensaco	FL	1098.13	
Web, Fred	555-2329	659 Safari	Palolt	CA	1.21	
Germane, M.	555-2330	1005 Rung	Patuxent	MD	555.12	
Unger, Oscar	555-2331	65A 5th St	Pasadena	TX	87.01	
Broth, John	555-2332	95 Vinceburn	Pinpoint	GA	2.23	↓

Figure 5-1. Example of a table.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.2.1.1.

5.2.1.1.1. *General*

+ In tables that contain more than seven rows, place a blank line or a subtle underscore after every fifth row. *Note:* This greatly improves readability by facilitating horizontal scanning (see Figure 5-1).

+ When the table has only limited display space available, but there are many columns that can possibly be displayed, either
(1) establish multiple views of the data and allow the user to select among them;
(2) give the user the capability to determine the data columns that show up in the table; or
(3) enable horizontal scrolling to display other columns. In this case, fix the key column (i.e., the leftmost column(s)) and scroll the other columns.

△ If two or more data items must be compared on a character-by-character basis, place one item directly above the other.

5.2.1.1.2.

+ For columnar data, place the label flush left above the column (without a colon) for alphanumeric data and flush right for numeric data that are anchored on a decimal point (see Section 4.3.1.2).

+ When numbering rows or columns or numbering items, start with the number "1" *not* "0."

+ Place units of measure (in parentheses) directly below the label or, if space permits, to the right of the label. For example,

Rate	Distance
(MPH)	(Miles)

5.2.1.1.3.

+ Display the reference (sometimes called the "key" or the "index") column as the leftmost column in the table. For instance, if the table is organized alphabetically, make the name column (sorted properly) the leftmost column. Display the rest of the columns from left to right according to their significance to the task.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

△ Construct the table so that it matches the organization the user requires or expects (e.g., because of a matching paper form).

△ If possible, do not separate columns that will be frequently compared.

5.2.1.2. Spacing and justification in tables

√ Numeric columns of data that do not contain decimal points must be right-justified; numeric columns of data that contain a decimal point must be justified with respect to a fixed decimal point. Maintain all significant zeros after a decimal point.

√ Alphanumeric columns of data must be left-justified; alphabetic must never be centered.

- + Use consistent spacing between columns and groups of columns.
 - Maintain at least three spaces between the longest entry in one column and the beginning of the next. *Note:* With proportional fonts, a "space" is the equivalent to the width of the letter "N."
 - When the columns are grouped, place at least five spaces between the groups. Intergroup spacing *must* be greater than intragroup spacing by at least two spaces (Figure 5-2).

+ Column spacing should be consistent from one display to the next, when similar data are being presented.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

January		February		Unit of measure
Name	Amount (\$)	Name	Amount (\$)	
Abbott	561.00	Adelson	23.00	↑
Anderson	25.00	Anderson	24.00	
Bischoff	1200.98	Caterow	600.00	
Bishop	706.78	Douglas	78.05	
Carlson	89.86	Folson	831.00	
Dickson	551.70	George	41.65	

Figure 5-2. Table spacing and justification.

5.2.1.3.

√ If numbering of items in a list exceeds the available display and the area is paged or scrolled, items must be numbered continuously from the first item in the display.

√ Maintain column and row headings when paging/scrolling through tables that extend beyond the available display area.

+ If users must compare items on the display that cannot be viewed in the same window at the same time, allow the user to split the window into several panes.

5.2.1.4. items in tables

See Section 5.1.5.

+ Provide distinctive coding to code items requiring user attention. Color and text attributes are effective when you need to code items in tables. For instance, important information might be presented in

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

boldface and unavailable information should be grayed out or presented in italics.

5.2.2. Data records

Data records display (view only) sets of labels and data fields, such as in Figure 5-3.

Database Query			
Client ID:	6789 A78Q		

Last Name:	Caver		
First Name:	William		
Company:	Spelunkers, Ltd.		
Address:	555 Main Street		
Mail Stop:	HND 1F326		
City:	Marion		
State:	IL	Zip:	68603

Phone:	708-565-9523		

Figure 5-3. An example of a data record.

5.2.2.1. General

√ When the same display is used for requesting data entry and for presenting a data record, use the same labels and sequence for each.

+ Make the layout of corresponding data fields consistent across displays. For example, do not present first name, last name in one display, then last name, first name in another display.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.2.3. Text

Often it is necessary to present long passages of text to the user. Be aware that reading times are significantly slower on a computer display than with hard copy. Below are presented some requirements and recommendations that improve reading times for computer-displayed text.

5.2.3.1. General

△ If users must read a substantial amount of text on-line, provide the ability to print out the specific page as well as the option of printing the whole document.

5.2.3.2.

√ Do not justify the right margin of computer-displayed text.

√ Separate all paragraphs with one blank line.

√ Use mixed upper and lower case. Do not use all upper case.

+ When presenting several paragraphs of text, make the display size of a text box between 4 and 17 lines in height. Ideally, use line spacings of 1.5 or 2 for text; avoid single spacing for long text passages.

+ For optimum reading speed, make line lengths between 50 and 78 characters. Continuous text should be displayed in *no fewer than* 50 characters per line. It is better to display a few long lines of text than several short lines. (In general, avoid using double columns of text on a computer display.)

+ Use headings and formatting to set off important sections of text. For instance, example (a) is strongly preferred

MANAGING LISTS

Method

To ADD a user to the list select the name from the list box in the left column and press the ADD button...

To CHANGE a user on the list select the name from

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

the list box...

over example (b)

TASK 1

Method

To ADD a user to the list select the name from the list box in the left column and press the **ADD** button...

To CHANGE a user on the list select the name from the list box...

Notice that in (a) the headings help set off the presentation of the text. The use of bold and formatting guide the eye through the page. In example (b), with every thing flush left it is difficult for the eye to find a good starting place or randomly pick out needed information.

5.2.3.3.

√ Do not hyphenate text.

+ Use normal punctuation.

+ Avoid the use of contractions. For instance, use "will not" rather than "won't."

+ Use short, simple affirmative sentences.

+ Use the active voice. It is much easier to read text written in the active voice than the passive voice.

√ When the text describes a sequence, the word order of the text must reflect the sequence of the task. For instance, "Enter Password before running applications" is better phrasing than "Before running applications, enter Password."

+ When a critical passage or word must be emphasized, highlight it by using one of the conventions (e.g., boldface) presented in Section 5.1.5.

+ Use headings to introduce and set readers' expectations for the text that follows the heading.

Δ Put the topic sentence of each paragraph at the beginning of the paragraph.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.2.3.4. Text lists

+ Display related items in a rather than in continuous text. For example, the following columnar representation:

States
Illinois
Indiana
Michigan
Ohio
Wisconsin

is easier to read than the paragraph representation: "States: Illinois, Indiana, Michigan, Ohio, and Wisconsin."

+ Generally, as long as scrolling is not required by the user, presenting a text list in a single column is better than multiple columns. For instance, it would be better to present a long list of 20 items than a two column list of 10 items each. But, not all lists can be presented in a single column.

+ When presenting a in more than one column, order the items vertically within each column. For instance, use this ordering:

A	E
B	F
C	G
D	H

rather than this ordering

A	B
C	D
E	F
G	H

(the tab order should follow the item ordering).

△ Adopt the user's logic when ordering the . If there are a lot of items to be ordered or if no other principle applies, order the list alphabetically.

5.2.4. Flowcharts

5.2.4.1. Purpose and use

Flowcharts are diagrams that illustrate sequential relations among elements or events.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Use flowcharts for schematic representation of sequence information.

5.2.4.2. General

+ Flowcharts should be presented in a logical order; that order will be (almost always) the sequence of steps required to perform some task.

+ In general, it is best to conform to established conventions for presentation of individual elements. For instance, if you are presenting a data flowchart use conventional shapes for elements as defined by national standards organizations.

√ When the user selects a displayed object to perform an action, highlight the item. In PICT-based graphics programs this is done by placing “handles” around the object. You might also highlight the object by using reverse video.

+ Users should be able to get detailed information about a particular flowchart object by picking the object and selecting a “more information” option on a menu, or perhaps just double-clicking on the object. This information should contain an explanation of what the object is, what it does and how it is used, as well as context-sensitive information.

5.2.4.3.

+ Lay out the flowchart so that it can be read from top to bottom and left to right.

+ If at all possible, adopt the convention that “yes” always leaves from the right side of the box and “no” always leaves from the left side or bottom of the box.

+ Only place one decision at each step. Do not combine multiple decisions into one decision box.

+ Use coding in flowcharts with a purpose, such as to distinguish different types of elements or elements out of normative range. For example, color might be used to demonstrate particular paths in the flowchart. Color is a good method of coding in flowcharts, but be careful not to use more than five levels of color coding. Because of the need for redundancy when using color coding, you should also alter the line/border width or use dashed rather than a solid lines.

5.3. User guidance

User guidance is on-demand information, such as "help," or system capabilities that aid the user's interaction with the application.

User guidance can be implicit or explicit.

Implicit guidance is integrated into the interface through well-designed menus, thoughtfully constructed client areas, and properties of screen objects. Much of this document is geared toward improving implicit user guidance.

Explicit guidance is available on request, explains the current object or command, and provides application information.

Explicit guidance is the subject of this section.

5.3.1. General

√ Allow the user to get help at any time, through either Help menus or HELP buttons.

+ Provide specific information relative to the task context rather than generic messages. For example, if the user entered information in the wrong format say "Format is MM/DD/YY" not "Invalid format."

+ State the consequences of an action before describing the action. For instance, say "To delete file, press Return" instead of "Press Return to delete file."

+ If a task has sequential steps, present the task in the order required to perform the task.

5.3.1.1. Style and grammar

User guidance that requires the presentation of large amounts of text should follow the guidelines in Section 5.2.3.

√ Do *not* use anthropomorphisms; that is, do not personify the computer. For example, instead of "I will continue when you press Return" say "To continue, press Return."

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ Words used in user guidance should be common, meaningful, unambiguous, and complete (i.e., no contractions or abbreviations). Avoid computer terminology, unfamiliar, or esoteric terms.

+ In general, use positively rather than negatively worded statements; negative statements, however, are appropriate for conveying exceptions to rules.

+ Use the active, rather than the passive, voice; use consistent grammatical construction; state messages in short, simple sentences.

+ Points (or facts) that must be remembered should be at the beginning of the message.

+ Place a period at the end of each sentence.

Δ Use drawings or pictures to illustrate text whenever possible.

5.3.2. Handling user errors

5.3.2.1. Preventing user errors

A well-designed application minimizes the number and severity of errors. The closer the application matches the user's task, the more likely that errors will be prevented. The use of interface objects like radio buttons, check boxes, and pop-up menus serve to prevent errors, because the application relies on the user to make a choice from pre-defined sets.

√ Require users to take some explicit action to enter information (e.g., on data forms with more than one required text box the user should have to press an OK or other affirmative button to complete the dialog.)

√ If the user takes some action that will cause the loss of contents of a current work display, prompt the user as to whether to save the current information, cancel the requested action or proceed without saving. (See Section 3.4.3 for common dialog boxes.)

Δ Make the application tolerant of common, predictable misspellings and typographical errors. For instance, if the user types CHICSGO, the application should suggest choices to the user as to the correct or intended entries without the user having to re-key the entry.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.3.2.2. Detecting and correcting incorrect entries

Errors include malfunctions to hardware or software and user inputs that are not recognized by the application.

Error detection is the application's edit checking capability; it tells the user that an error has been found and the conditions that caused the error and effects of the error, if possible.

Error correction is the application providing information on the cause of the error and recovery procedures. For instance, the application may require the user to re-enter the name of a file if the application cannot find a file by the user-supplied name.

√ When an error is detected and requires the user to change information, all previous information that was correct must remain intact. In other words, the user must never be forced to re-enter previously correct information.

√ If a user's action may have unintentional or destructive consequences, such as loss of data, present the user with a warning or confirmation message making the user confirm or cancel the action before executing the command.

+ Detect errors immediately after an explicit OK, Send or other appropriate action is taken by the user to dismiss the error dialog box.

+ When the application detects an error present an error dialog box describing the error; post the dialog box, if at all possible, in such a way that it does not obscure the item(s) that caused the error. When the user confirms that he has read the message (by pressing OK), the dialog box disappears and the application highlights the offending field(s) and the cursor is placed in the left-most and top-most field.

5.3.3. Messages

This section addresses the variety of messages, status information, and user queries.

5.3.3.1. Error messages

√ Allow the user to access help directly from an error message.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

√ The documentation must include a listing and description of all error messages.

- + Make error messages:
 - Brief, but informative.
 - Convey what is wrong, where the error is, and what corrective actions can be taken.
 - Provide information specific to the task rather than generic (e.g., "Invalid data" is a non-specific, unhelpful error message).
 - Non-judgmental; do not accuse or patronize the user; do not personify the computer or attempt humor.
 - Go away as soon as the error is corrected or the user dismisses the error dialog box.

- + When successive user actions result in the repetition of the same error message, re-phrase the error message for the third and all subsequent times. Additionally, for the third repetition of the error message, the application should beep twice before presenting the error message.

- + If the set of valid inputs is small, present the set of alternatives with the error message.

- + When the error involves a logical unit of input (e.g., a text in a text box), the offending entry should be highlighted when the dialog box is removed.

- + If there are multiple errors, tell the user there are multiple errors. All fields containing errors should be simultaneously identified to the user.

5.3.3.2. Information area

The **information area** is a read-only, non-scrolling display located at the bottom left of a primary window where non-essential, informative messages appear (*before* the user actually selects the object or action) about an object or a selection. For example in Microsoft Excel, if the user posts the "File" menu and passes the pointer over (but does not select) "Open" the information area presents "Open saved document."

- + Most primary windows should have an information area.

√ Use the information area for information that the user is not *required* to see. It should not be used for critical information -- use a dialog box instead.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

+ When the cursor is on a command (a menu item or button), use the information area to present the results of the user action if the command is selected.

+ When the cursor is on an object, use the information area to suggest the default action, the most appropriate action, or how to perform available actions.

+ When the cursor is on a list box, use the information area to tell the user how many items they can select, e.g., "Select one item" or "Select as many as apply."

+ Pass messages to the user about the normal completion of an action in the information area (e.g., *file* was successfully saved).

+ Allow the user to turn the information area on and off through a menu choice (e.g., under the "Windows" menu; see Section 3.4.2).

+ Remove the message from the information area when it is no longer relevant.

More reading

IBM SAA CUA AIDR, p. 119-120.

5.3.4. On-line help

On-line help provides directions, explanations, and information to the user:

- directions for: executing a command, accessing and using an application's capability, accomplishing a specific goal, and handling error conditions.
- explanations of concepts associated with a task or interface components, and commands or dialog components and the actions they perform.
- information on the syntax required to do a task, a listing of available commands or options, and a statement of valid ranges for a field.

Users access help by simple consistent actions: in Apple by pressing the HELP key or COMMAND-?; in Windows by pressing the F1 key or the SHIFT+F1 keys; and in Motif by pressing the F1 key.

In general, users will come to help in one of two instances. First, users will request help when they have a specific problem and want a specific context-sensitive answer. Alternatively, many users consider help a safe and efficient means of learning the application. Thus, users will browse help, just to get a feel for the application, with no specific goals

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

or problems in mind. The primary purpose of on-line help, however, is to provide context-sensitive solutions to user problems.

Context-sensitive help is information based on the location of the pointing cursor on the screen, the active application, the current step in a dialog, most recent error message, etc. Context-sensitive is applicable when:

- Users need to request help that is specific to a current task or object.
- The task has sequential steps.

Facilitating **browsing help** is done by presenting the user with a help index; the user enters a help topic or picks from a list of available topics. Browsing is applicable when:

- There is no context available for context sensitive help for a selected object.
- Users may have trouble accurately specifying the help topic without assistance.

√ Users must be able to initiate a help request whenever they want, get help on any topic they desire, control the type of help information, and exit help at any time.

+ It is *strongly recommended* that on-line help remain visible until the user explicitly removes it.

5.3.4.1. Style

+ Keep the writing simple, concrete, and natural.

+ Provide information that focuses on the task -- include all needed information, be sure all information is correct, and exclude anything that is not needed. The best way to ensure that help is helpful is to review and test help with target users.

+ Use a lot of white space. In general, no more than 40% of the pixels should be occupied by words and figures.

△ Anticipate the reasons for the confusion and how the help text will address that confusion. Write on-line help with the user community and the specific application tasks in mind. Users often access help because they are unsure about something.

△ If the users can enter a term for which they want help, the application should accept (non-technical) synonyms and close spelling matches.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

5.3.4.2. Content

+ The following headings and descriptions are strongly recommended for and window-level help:

- *Purpose* - Begin explaining the purpose of the screen or field, and when and how the information on this display fits into the business purpose.

Follow this description with descriptive and procedural information as required by the task.

- *Concept* - Use the *concept* heading only for complex or confusing fields, such as "Internal wiring:"; it does not seem likely that a concept statement for "Street Address," for instance, would be needed. For example, does the application require information to be entered in a particular order or is certain information pre-populated and where does the application get the information to populate the display?
- *Syntax* - Explain to the user all the elements of the display in terms of the computer objects (fields, lists, etc.). Answer questions like: How do I get out of this display?

Describe the syntax or the form of the entry that is required in fields. Also report on the edit checking that the application does on this field. One reason users access field-level help is because an edit check has flagged the field. Help should tell the user how to correctly enter information so that the user is capable of entering the information properly.

- *Examples* - Provide examples of how commands or functions are used. Use concrete examples to explain abstract topics. Examples are one of the most important parts of help.
- *More Information* - Provide cross references to closely related topics that can be accessed through on-line help. *Note:* on-line help must sufficiently explain the concepts because the manual may not be available when the user requests help.
- *Notations* - If at all possible, provide an editable field with each help topic for users to make annotations for their own reference later on.

5.3.5. Feedback, prompts and cues

Feedback is the perceptible response from the application that results from user actions. Feedback indicates the current state of system hardware or software, such as available applications, modes, processes, etc.

Examples:

- The tracking of the pointer when the mouse is moved.
- "Shrinking" a document into an icon when the user "Minimizes."
- Removing the drop-down menu when the user selects a menu item.

√ Every user action should produce perceptible feedback from the application. Most interface objects have feedback built into their usage.

√ The application must provide feedback (e.g., a message dialog box) must be provided when:

- there is a 2 to 15 second delay in processing a command entry -- change the cursor shape (see Section 3.2.1.).
- an operation will take 15 seconds or longer -- provide a progress indicator dialog box.
- the processing time delay is unknown -- change the cursor shape (see Section 3.2.1.).
- users have minimal training or are infrequent users of the application -- use the information area liberally.

√ Show all characters (and control characters that are ignored) that the user types on the display, except for passwords and for other security reasons.

+ It is strongly recommended that the application anticipate system failures and report them to users; therefore, provide a dialog box telling the user of the impending failure (e.g., "Running out of memory-Save and close windows now").

+ Following an application interrupt, provide feedback assuring the user that the application is still working and returned to its previous state.

+ Make system responses fast. Target times are as follows:

- Movement between fields on the same screen, less than 250 msec.
- Cursor tracking following mouse movement, less than 50 msec.
- Keystrokes in a text box, less than 100 msec.

+ Provide status and error information on requests sent to a remote machine, e.g., printer or database retrieval.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

△ Feedback should be non-intrusive and not detract from the task. For instance, the pointer change in text entry or periods of delay are not intrusive and blend directly into the task.

△ If possible, make feedback accommodate the skill level of user. Novice users need more qualitative feedback than experts.

△ Integrate feedback into how users work with the task. For example, if the user is likely to be looking away from the display, and the application needs attention, the application should beep.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

6. Porting to GUI Platforms

6.1. GUI to GUI

To be completed in a future release.

6.2. Non-GUI to GUI

To be completed in a future release.

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

7. References and index

7.1. References

Reference	Short Form
Apple Computer, Inc. (1987). <i>Human Interface Guidelines: The Apple Desktop Interface</i> . Reading, MA: Addison-Wesley.	Apple HIG
Galitz, W.O. (1985). <i>Handbook of Screen Format Design</i> . Wellesley Hills, MA: QED Information Sciences.	Galitz
Gould, J. (1988). How to design usable systems. In M. Helander (ed.) <i>Handbook of human computer interaction</i> , (pp. 757-789). New York, NY: North Holland.	Gould
International Business Machines Corporation. (1989). <i>Common User Access Advanced Interface Design Guide</i> . SC26-4582-0.	IBM SAA CUA AIDG
International Business Machines Corporation. (1991). <i>Common User Access Advanced Interface Design Reference</i> . SC34-4290-00.	IBM SAA CUA AIDR
International Business Machines Corporation. (1991). <i>Common User Access Guide to User Interface Design</i> . SC34-4289-00.	IBM SAA CUA GUID
Kobara, S. (1991). <i>Visual Design with OSF/Motif</i> . Reading, MA: Addison-Wesley.	Kobara
Lund, A. (1991). <i>Ameritech Services Working Document on Human Interface Standards</i> .	Lund
Microsoft. (1992). <i>The Windows Interface: An Application Design Guide</i> . PC28921-0692.	Windows ADG
Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In <i>Proceedings of CHI'90</i> Seattle, WA,	Nielsen

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

April 1-5, 1990) ACM, New York, pp. 249-256.

- | | |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| Norman, D. (1988). <i>The Psychology of Everyday Things</i> .
New York, NY: Basic Books. | Norman |
| Open Software Foundation. (1991). <i>OSF/Motif™ Style
Guide</i> . Englewood Cliffs, NJ: Prentice Hall. | OSF/MSG |
| Shneiderman, B. (1986). <i>Designing the User Interface:
Strategies for Effective Human-Computer Interaction</i> .
Reading, MA: Addison-Wesley. | Shneiderman |
| Smith, S.L., & Mosier, J. N. (1986). <i>Guidelines for Designing
User Interface Software</i> . Bedford, MA: Mitre
Corporation. Report 7 MTR 10090, ESD-TR-86-278.
Available from: National Technical Information
Service, Springfield, VA. | Smith |
| Sun Microsystems, Inc. (1990). <i>OPEN LOOK™ Graphical
User Interface Application Style Guidelines</i> .
Reading, MA: Addison-Wesley. | OL/ASG |
| Tullis, T. (1988). Screen Design. In M. Helander (ed.),
<i>Handbook of Human-Computer Interaction</i> . New
York: Elsevier Science Publishers (North-Holland). | Tullis |

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

7.2. Index

Always on top menus 68
Abbreviations 127, 136, 156, 157, 172
About <application name> 93,94, 99
Accelerators 12, 66, 67, 82, 85, 88, 90, 92,
93, 127
Acronyms 127, 136
Action-object model 27, 28
Actions, defined 25
Active voice 127, 168
Active window 35, 45
Alert 79, 87, 184
Alphanumeric fields 139
Alt 69, 73, 82, 85, 127
Alt+backspace 88
Ameritech signature 81
Ameritech standard menus 81
Amount of information 142, 148
Application modal secondary windows 35,
36
Arrays 184
Arrow head 65
Arrow heads 65
Assumptions 10
Attribute groups 66
Attribute menu 104, 106, 119, 120, 122
Attribute menu items 103, 104, 106, 112,
119, 121, 122
Audience, intended 9
Beachball cursor 74
Benefits for ameritech 8
Benefits for designers 8
Benefits for users 8
Binary choice 102, 116, 123, 184
Binary commands 111, 112
Blink coding 158
Blinking 158
Blue 79
Boldface 158, 168
Brightness 158
Browsing help 176
Button 37, 42, 44, 48, 49, 50, 53, 54, 56, 57,
58, 59, 60, 69, 106, 113, 122, 125,
126, 127, 128, 129, 130, 132, 137,
145, 151, 156
Button size 129
Buttons, see also command buttons 69
Capitalize 127
Cascade indicator 65
Cascaded menu 61, 106
Caution 74
Character properties 98
Check boxes 70, 116, 117, 119, 122, 123,
124, 150, 172
Check buttons, see also check boxes 70
Check mark 66
Checked fields 139
Checkmark 120
Clear 88, 90, 113, 127, 185
Client area 36, 37, 38, 40, 42, 43, 44, 45,
46, 48, 61, 69, 78, 86, 87, 89, 90,
125, 156
Clipboard 88, 89, 90, 131
Close 55, 82, 84, 85, 87
Close box 39, 40
Close dialog box 52, 55
Code 165
Coding 78, 151, 157, 158, 159, 160, 165,
170
Color 78, 79, 115, 118, 159, 165, 170
Color coding 78, 159, 170
Color palettes 72, 99
Column orientation 135
Column spacing 164
Columnar data 163
Combination box 71, 184
Combination boxes 71
Commands and an attributes, differences
112
Command area 43

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- Command button 53, 56, 57, 58, 113, 142, 184
- Command buttons 59, 60, 69, 104, 127, 130, 133, 151
- Command completion dialog boxes 38, 48, 53, 65
- Commanddialog 108
- Commands 22, 43, 51, 53, 58, 61, 63, 67, 68, 72, 73, 76, 85, 90, 103, 104, 106, 107, 108, 110, 111, 112, 119, 121, 123, 125, 126, 127, 128, 130, 131, 132, 175
- Common interface objects 99
- Conceptual consistency 19
- Contents 93
- Context sensitive help 176
- Contractions 136
- Control 59, 63, 69, 71, 72, 89, 109, 117, 148, 151, 176, 184, 185
- Control menu 37, 38, 47, 55, 81, 82, 83, 84, 85, 184
- Control menu, primary windows 81
- Control menus, secondary windows 85
- Copy 88, 89, 137, 139, 180, 185
- Cross hair 74
- Ctrl 77, 82, 85, 88, 127
- Ctrl+ insert 88
- Ctrl+c 88
- Ctrl+v 88
- Ctrl+z 88
- Cues 177
- Cursor 37, 67, 68, 69, 73, 74, 75, 77, 89, 106, 128, 139, 145, 151, 152, 153, 174, 175, 178, 184
- Cursor navigation 73
- Cursor shapes 73
- Cursor shapes, special 73
- Cut 88, 89, 131, 137, 180, 184
- Data columns, alphabetic 164
- Data columns, numeric 164
- Data fields 165, 166
- Data forms 70, 143, 146
- Data groupings 156
- Data input objects, defined 25
- Data output objects, defined 25
- Data records 165
- Data-centered 14
- Dates, standard format 99
- Default button 49, 50, 58, 129, 153
- Default buttons 49, 58, 129
- Delimiter characters 137
- Delimiters 137
- Dependencies among controls 151
- Design for error 20
- Design heuristics 17
- Desktop 42, 83, 184, 185
- Desktop metaphor 32
- Detecting and correcting incorrect entries 173
- Dial 184
- Dialog box 48, 52, 53, 54, 55, 58, 59, 60, 65, 87, 94, 97, 126, 128, 129, 139, 140, 141, 143, 145, 146, 148, 150, 151, 152, 153, 184, 185
- Dialog box command buttons 129
- Dialog boxes 35, 41, 48, 51, 52, 58, 60, 94, 108, 121, 123, 129, 142, 148, 149, 150, 151, 152, 185
- Dialog boxes, general recommendations 58
- Dials, in apple - see also sliders 72
- Diamond 66
- Dimmed 67
- Dimming 126, 131
- Direct manipulation 31
- Disabled 63, 67, 68, 86, 89, 90, 108, 131
- Disabled commands 67
- Disabling 126, 128
- Display output, general 155
- Display size 151
- Document conventions 12
- Drag and drop 31
- Drop-down combination box 71
- Drop-down lists 71, 103, 106, 107
- Edit checks 145, 146
- Edit menu 88
- Ellipsis 48, 64, 65, 86, 126
- Enabled 41, 44, 68, 83, 151
- Enabled commands 67
- Entry fields 134, 139, 140, 142
- Entry fields, see also text boxes 70

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- Error correction 173
- Error detection 173
- Error dialog box 50-52, 58, 79, 146, 174
- Error messages 142, 173, 174
- Errors 50, 116, 130, 146, 158, 172, 173, 174
- Exit 85, 87, 176
- Expert techniques 29
- Explicit guidance 171
- Explicit operation model 26
- Feedback 52, 54, 67, 125, 137, 177, 178, 179
- Field alignment 142
- Field types 139
- Field-level help 177
- File menu 85
- Fileselectiondialog 48, 108
- Find and replace 90, 98
- Fine-position selections 74
- Fixed palettes 114
- Fixed-length text entries 138
- Floating palettes 38, 68, 113, 114, 126, 157
- Flowcharts 169, 170
- Focus 76
- Font sizes 160
- Fonts, coding 159
- Format aids 134, 139, 140
- Function key 67
- Getting field input 134
- Glossary 185
- Graphical control 72, 104, 115
- Graphical controls 115
- Graphics 107, 115, 118, 150
- Grayed out, menu items 63
- Gricing 126, 131
- Green, color 79
- Group box 128
- Group titles 151
- Grouping controls 150
- Grouping heuristics 150
- Grouping techniques 142
- Grouping techniques, see also data groupings 156
- Grouping buttons 128
- Headings 44, 142, 143, 144, 152, 155, 157, 159, 160, 165, 168, 177
- Help 51, 53, 54, 55, 56, 57, 58, 60, 91, 92, 94, 128, 139, 142, 146, 157, 171, 173, 175, 176, 177
- Help buttons 171
- Help menu 92, 93, 94, 171
- Help menu, optional items 94
- Help, browsing 176
- Help, context-sensitive 176
- Help, cursor for 74
- Hide 83, 184
- Highlighting 78, 141, 151, 157, 158, 159, 160
- Highlighting and coding, tables 165
- Horizontal scanning 163
- Hourglass 74
- Hourglass cursor instead of the watch cursor 74
- How to Use Help 93
- Hyphenations 136
- I-beam 74, 75
- Icon lists 104, 113
- Icon palettes 113
- Icon, maximize button in Motif 41
- Icon, minimize button in Motif 42
- Icon, restore button in Motif 42
- Iconize 83
- Iconize, use minimize 44
- Icons 100, 104, 113, 114, 132, 133
- Implicit guidance 171
- Implicit operation model 26
- Inactive windows 45, 46
- Indent 143
- Indicator 64
- Indirect manipulation 31
- Information area 40, 43, 45, 128, 142, 152, 174, 175
- Information dialog boxes 48, 49, 54
- Information displays 162
- Input focus 45, 185
- Insertion point 105, 120, 184
- Interaction styles 31
- Italics 159
- Justification 137, 139, 143, 163, 164, 167
- Keyboard 10, 58, 64, 66, 69, 73, 75, 76, 127, 130, 137, 153

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- Keyboard accelerators 67
- Keyboard commands, for switching windows 47
- Keyboard equivalents 127
- Keyboard focus 152
- Keyboard navigation 75
- Keyboard selection 75, 76
- Keyboard selection of menu items 127
- Labeling 126, 128, 131, 148
- Labels 48, 54, 64, 69, 89, 113, 121, 125-137, 139, 140, 142, 143, 147, 155, 156, 157, 159, 160, 263, 165, 166
- Labels, buttons 126, 129
- Labels, data forms 147
- Labels, entry fields 134
- Labels, on icons 113, 133
- Labels, tables 163
- Layout 48, 141, 147, 148, 149, 151, 152, 166
- Layout, flowcharts 170
- Layout, table 162
- Layout, text 167
- Left justify 142, 147
- Left-pointing arrow 74
- Legend 157
- Lists 64, 70, 71, 84, 92, 105, 107, 108, 109, 113, 118, 121, 122, 130, 165, 169, 184, 185
- List boxes 70, 71, 76, 106, 109, 110, 121, 122, 150, 175
- Locating controls 151
- Location cursor 73, 75, 152
- Logical groupings 142
- Logo 81
- Lower 40, 82, 84, 142
- Lower case 127, 137
- Maximize operations 42, 44, 82, 83, 84
- Maximize button 41, 42, 43, 44
- Measurement unit 140
- Menu 42, 44, 61, 62, 63, 64, 65, 66, 68, 69, 74, 81, 82, 86, 87, 88, 89, 90, 91, 92, 105, 106, 111, 113, 125, 127, 130, 131, 132, 151
- Menu bar 42, 44, 61, 63, 68, 69, 81, 85, 88, 90, 91, 92, 106, 113, 126, 142
- Menu bar, activating from the keyboard in Apple 69
- Menu bars 42, 44, 63
- Menu group 131
- Menu items 12, 48, 61, 63, 64, 65, 66, 67, 68, 69, 75, 81, 84, 87, 89, 90, 104, 105, 111, 118, 119, 125, 126, 127, 128, 130, 131
- Menu title 63, 64, 85, 130, 131
- Menus 12, 37, 42, 61, 62, 63, 64, 67, 69, 105, 106, 113, 114, 126, 127, 130, 132
- Message area 43, 184
- Message bar 45
- Message dialog boxes 52, 54, 55
- Minimize operations 42, 44, 83, 84, 153
- Minimize button 42, 44
- Mnemonics 64, 69, 82, 85, 88, 90, 92, 93, 127, 130
- Modal operations 35, 48, 49, 50, 57, 58, 98
- Modal dialog boxes 22, 142
- Modal secondary windows 35
- Modes 22, 73, 184
- Modeless operations 35, 48, 55, 56, 131
- Modeless dialog box 146
- Modeless secondary windows 36
- Mouse 10, 76
- Mouse button 68, 77
- Mouse pointer 68
- Mouse-based selection 76
- Moving windows with the title bar 47
- Multiple contiguous selection 76
- Multiple discontinuous selection 76
- Multiple selection 77, 184
- Multiple-choice component 185
- Multiple-selection lists 119, 121
- Mutually exclusive attribute groups 65, 66, 70, 102, 103, 104, 115, 124
- N of many 66, 118, 123
- Navigating among windows 47
- Navigation 73, 76, 126, 127, 130, 145, 148, 152
- Navigation, menus 68
- Navigation, tables 164
- Navigation, within a window 46

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

- Non-mutually exclusive attribute group 66
- Non-mutually exclusive choices 118, 119, 121, 122, 124
- Non-mutually exclusive selection 70
- Numeric fields 139
- Object-action model 27
- Object-based interaction 24
- Objects, defined 24
- Ok/cancel dialog box 52, 53
- On-line help 175-177
- One of many 65, 102, 106, 116
- Operations, defined 25
- Option buttons, see also radio buttons 70, 109
- Option menu 184
- Option menu, in Motif - see also pop-up menus 72
- Optional elements 39, 41, 43, 66
- Optional elements, Apple window 39
- Optional elements, Motif window 41
- Optional elements, Windows' window 43
- Options >> 60
- Orange, color 79
- Organization, tables 163
- Organizing controls 148
- Osca™ 23
- Output to the user 11, 154
- Overlap 90, 91, 92
- Paging 152, 165
- Palette 104, 106, 114, 118
- Paper forms 147
- Passive voice, avoid 168
- Paste 88, 89, 90, 137, 139
- Placement and location of buttons 129
- Placement of display elements 140
- Plus sign cursor 74
- Pointer 47, 67, 68, 74, 77, 79, 104, 113, 178
- Pointing cursor 73, 75, 176, 184
- Pop-up menu 72, 103, 106, 107, 108, 110, 114, 125, 150, 172
- Position cursor 73
- Preventing user errors 172
- Primary windows 34, 38, 41, 42, 43, 44, 47, 61, 63, 81, 83, 86, 87, 90, 91, 92, 142, 174, 184, 185
- Print 85, 87, 97
- Printer error 98
- Printing 97
- Printing dialog box 98
- Progress indicator 178
- Prompts 177
- Property dialog box 52, 53, 54
- Proportional scroll boxes 47
- Proprietary markings 81
- Protected fields 139, 145
- Pull-down menu 42, 44, 61, 63, 102, 104, 106, 118, 119, 126
- Push buttons, see also command buttons 44, 69
- Question 56, 60, 74
- Question dialog box 52, 56
- Quit 85, 87
- Radio button 70, 103, 109, 110, 124, 150, 172
- Readability 147, 159, 163
- Reading speed 167
- Recommended colors 78, 79
- Red 79
- Redo 53, 54, 88, 89
- Required fields 137, 139, 145
- Requirements conventions 12
- Resize 37, 40, 74, 83, 152
- Resize box 37
- Restore 42, 44, 82, 83, 85
- Restore button 41, 42, 44
- Reverse video 67, 139, 158
- Right justify 142, 147
- Right-pointing arrow 74
- Routing choice 184
- Row orientation 135
- Sash 42, 184
- Save 59, 60, 85, 86, 87, 95, 172, Save as dialog box 85-87, 95, 96
- Save changes 96
- Save changes dialog box 84, 86, 97
- Scale, in Motif - see also sliders 72
- Scope 9
- Screen density 155
- Scroll bar 40, 43, 45, 46, 47
- Scroll bars 40, 42, 44, 45, 46, 47, 108, 138

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Scroll box 46
Scrolling 152, 165
Scrolling indicator 65
Search for help on 93
Searching text or lists 162
Secondary windows 35, 38, 41, 42, 44, 47,
48, 83, 84, 85, 87, 142
Section headings 143
Select all 88, 90, 152
Select and operate paradigm 27
Selecting commands 127
Selecting fields 74
Selection 73, 74, 75, 76, 77, 89, 90, 104,
109, 118, 125, 127
Selection of default buttons 129
Selection, accidental 131
Selection, single and multiple 76
Selectiondialog 108
Separating lines 138, 139
Separator 61, 65, 84, 86, 87, 89, 90, 92,
129, 130, 137
Sighting 74
Signal frequencies 80
Simplicity 17
Single selection 77, 184
Single-choice component 185
Single-selection lists 103
Size command 40, 41, 43, 44, 47, 82, 83,
85, 115, 151, 152
Size box 40
Size of display 152
Slider 72, 115
Social security number, standard format 99
Sound 79, 80, 150
Source documents 147
Spacing between groups 150
Spin button 109
Split 40, 44, 165
Split bar 40, 44
Split box 44, 184
Standard controls 69
Standard dialog boxes 51, 94
Standard key assignments 127
Standard menus 42, 126
Standard menus, Ameritech 81
Status bar 45
Style and grammar 171
Style guides 9
Sub-headings 144
Sub-section headings 143
Subtle lines 65, 128, 141, 150
Switch to... 82, 84
System modal secondary windows 35, 36
System-defined menu items 127
Tab 73, 152
Tab order 76
Tabbing 145
Table 104, 152, 162, 163, 164, 165
Target area 128
Tear-off menus 61, 68, 113
Telephone numbers, standard format 99
Terminology conventions 13
Text 6, 60, 66, 70, 73, 74, 75, 79, 89, 118,
123, 137, 138, 139, 151, 152, 162,
166, 167, 168, 172
Text box 71, 134-147, 152, 167, 172, 178
Text entry 134, 140, 141, 143
Text insertion cursor 74
Text list 168, 169
Tile command for windows 90, 91, 185
Times, standard format 99
Titles 36, 38, 39, 54, 63, 64, 131, 127, 151,
152, 155, 160
Title bar 37, 38, 39, 40, 41, 42, 43, 44, 45,
82, 85
Titling buttons 128
Toggled command buttons 112, 113
Toggled-menu items 103, 111, 112, 113
Toolbox 113
Touch area 128
Trademarks 2, 81
Truncatation 138
Tutorial 93
Unavailable items 63
Unchecked fields 139
Underlining 151, 160
Undo 53, 54, 60, 88, 89, 128, 131, 137, 162
Units of measure 136, 137, 140, 163
Upper and lower case characters 159, 162,
167

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES

Upper case characters 159
Usability testing, see also user testing 15
Use of color 78
Use of sound 79
User guidance 43, 45, 142, 146, 171
User input controls, defined 25
User interface architectures 23
User testing 14, 15, 16, 23, 157
User-centered design 14
Using this document 9
Visual angle 156
Visual clutter 111, 135, 139, 155
Volume, sound 80
Walking menus 61
Warning boxes 49, 50, 52, 57, 79, 126
White space 128, 150, 176
Widgets 47, 48, 108, 121
Windows 34-48, 60, 61, 63, 74, 75, 81, 83,
84, 86, 90-92, 113, 126, 129, 141,
142, 145, 146, 148, 150-153, 165
Window elements 36, 37
Window hierarchy 84
Window list 47, 184
Window menu 36, 37, 45, 47, 64, 90, 91, 92
Window names 92
Window, active/inactive 45
Window-level help 177
Wording and style, text 168
Working dialog box 52, 56, 57
Workspace 34, 184
X Pointer 74
Zoom box 40, 184

AMERITECH

STANDARDS FOR INFORMATION SYSTEMS

Published by Ameritech Services, Inc.

IS: 14-100

TOPIC: ARCHITECTURE RELATED STANDARDS

SECTION: AMERITECH GRAPHICAL USER
INTERFACE STANDARDS AND
DESIGN GUIDELINES
