MICROSOFT® WINDOWS™

Microsoft Professional Reference

Covers Windows 95 & Windows NT™!

软件设计的 Windows界面指南

张丽霞 孟丽青 潘旭燕 等 译
张冬梅 贾银潇 等 审校

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

Microsoft® Press

软件设计的 Windows 界面指南

〔美〕Microsoft Corporation 著

张丽霞 孟丽青 潘旭燕等 译

张冬梅 贾银潇等 审校

电子工业出版社

内 容 提 要

本书将指导读者为运行于 Microsoft Windows 操作系统中的应用程序创建风格良好、可视的并且功能一致的用户界面。本书的内容是由美国微软(Microsoft)公司提供的。对于所有使用 Windows 最新版本进行工作的编程人员和设计人员来说，无论他们的经验是否丰富或者使用哪一种开发工具，本书都可以说是他们一本最基本的参考手册。

在本书中，介绍了用户界面设计的基本原理以及设计方法，并且说明了以数据为中心的概念(比如对象和属性)怎样才能够应用到界面设计中去。还为读者提供了有关鼠标、键盘、笔交互作用以及怎样利用由系统提供的通用界面的细节性信息。这本书同时也考虑到了某些特殊的问题，并且对这些问题进行了阐述，这些特殊的问题包括网络计算、国际用户以及残疾用户等等。

本书的主题包括常规输入技术、窗口、菜单、控件和工具栏、Microsoft OLE、用户帮助、系统集成、可视设计等内容。

如果读者正在考虑为 Microsoft Windows 开发应用程序，那么本书可以说是一种非常重要的信息资源。

软件设计的 Windows 界面指南

〔美〕Microsoft Corporation 著

张丽霞 孟丽青 潘旭燕等 译

张冬梅 贾银潇等 审校

责任编辑：胡毓坚 特约编辑：刘青

*

电子工业出版社（北京市万寿路）

电子工业出版社发行 各地新华书店经销

北京市顺义县天竺颖华印刷厂印刷

*

开本：787 × 1092 毫米 1/16 印张：26.75 字数：651 千字

1996 年 6 月第 1 版 1996 年 6 月第 1 次印刷

印数：5000 册 定价：45.00 元

ISBN 7-5053-3547-2/TP·1439

著作权合同登记号

图字：01-1996-615

译者序

Microsoft 公司推出的 Microsoft Windows 3.X，Windows 95 及 Windows NT 在计算机界被广泛地作为应用平台工作，编写在 Windows 平台上的应用程序就成为一件很普遍的事情。

本书将指导读者为运行于 Microsoft Windows 操作系统中的应用程序创建风格良好、可视的并且功能一致的用户界面。本书的内容是由美国微软(Microsoft)公司提供的。对于所有使用 Windows 最新版本进行工作的编程人员和设计人员来说，无论他们的经验是否丰富或者使用哪一种开发工具，本书都可以说是他们一本最基本的参考手册。

在这本书中，我们介绍了用户界面设计的基本原理以及设计方法，并且说明了以数据为中心的概念(比如对象和属性)怎样才能够应用到界面设计中去。我们还为读者提供了有关鼠标、键盘、笔交互作用以及怎样利用由系统提供的通用界面的细节性信息。这本书同时也考虑到了某些特殊的问题，并且对这些问题进行了阐述，这些特殊的问题包括网络计算、国际用户以及残疾用户等等。

本书的主题包括：

- 常规输入技术——漫游、选择、查看、编辑和创建，同时包括命令和直接操纵方法，比如拖动和放置(拖放)等等。
- 窗口——主要和从属类型以及它们的组件，其中包括属性清单、对话框、消息框、调色板窗口以及弹出式窗口等等。
- 菜单、控件和工具栏——介绍类型和它们的组件，以及何时使用它们。
- Microsoft OLE——针对 OLE 嵌入与链接对象、可视编辑以及其他的活动形式，怎样为设计界面提供支持。
- 用户帮助——怎样使用帮助的上下文形式，其中包括工具提示和 Wizard。
- 系统集成——设计自己的软件，让它的界面功能和操作方式和 Windows 保持一致。
- 可视设计——对颜色、布局、字模和图形合理、有效地应用。

如果读者正在考虑为 Microsoft Windows 开发应用程序，那么本书可以说是一种非常重要的信息资源。

参加本书翻译工作的有贾银潇、张冬梅、张丽霞、孟丽青、潘旭燕、黄世枫、成燕、吴旋、余新、陈冬梅、王明等同志，张冬梅、贾银潇等对本书进行了认真的审校，尤晓东进行了全书的统编以及编审工作。

由于时间仓促，翻译过程中难免出现错误，欢迎广大读者指正。

译　者

1995.10 于北京

简 介

欢迎您阅读本书，该书是设计运行于 Microsoft Windows 操作系统中的软件时一本不可缺少的参考指南。与其他任何内容相比较言，软件界面的设计在整个软件的设计过程中是最重要的，它将严重影响用户对一个软件产品的感受。本书为读者介绍了怎样设计出具有良好界面、可视化和功能性比较强的 Windows 应用程序。

新增内容

本书连贯地介绍了与 Microsoft OLE 有关的内容。OLE 是 Windows 用户界面设计历史中一种质的飞跃，它代表了在程序设计过程中的新进展。无论是基本的和图形化的对象，还是其他更多的对象，它们的设计方法都是崭新的。这就是说，无论是以应用为中心的界面，还是以更多数据为中心的界面，它们的设计都由 OLE 提供了强有力的支持。相应地，现在的程序开发者和设计者都需要重新考虑一下自己软件的界面问题——包括基本的组件和对应的操作，以及应用于它们的属性。这一点是非常重要的，从一个用户的角度来看，计算机中运行的应用程序正在变得越来越多，而应用程序的重要性正在变得越来越小，同时在这些应用程序界面对象后面所隐藏着的那些内容却正在变得越来越多。在没有必要考虑应用程序的前提下，用户现在可以和数据进行交互作用，这样就可以把自己的精力集中于具体的任务上面，而不是对不同应用程序的操作上面。

在改编现成的、以 Windows 为基础的软件的时候，请读者确保自己已经考虑到下面几个重要的方面：

- 标题栏和图标
- 属性夹
- 传输模式(包括“拖动和放置”)
- 弹出式菜单
- 新控件
- 和系统的集成
- Help 界面
- OLE 嵌入和 OLE 链接
- 窗口、控件、图标和可视化设计
- 窗口管理
- 最小化窗口的设计

在以后的章节中，我们将深入介绍上述每一种元素的情况。

怎样使用本书

本书是为那些基于 Windows 应用程序软件的开发人员和设计人员而编写的。对于那些有兴趣更好地理解 Windows 环境和 Windows 所支持的人机界面原理的人们来说，这本书同样也是适用的。本书的内容涉及到下列的几个领域：

- 基本的设计原理和进程——基本的设计理论、有关人类行为的假定、设计方法学以及界面的具体概念。
- 界面元素——与界面中不同组件有关的描述性信息以及怎样使用这些组件。
- 设计细节——在使用界面元素的时候，为了让自己的设计和风格能够给人留下深刻的印象而需要注意的一些问题。
- 附加信息——总结和快速参考信息、一个参考书目、一个为了援助产品本地化而提供的世界各种语言术语对照清单以及一个术语词典。

本书侧重于介绍应用程序用户界面的设计和元素。尽管偶尔介绍了一些技术参考信息，但是这本书的主旨并不是为读者提供有关技术实施或者应用程序编程接口(Application Programming Interfaces，API)的细节信息，这是由于现在存在着多种类型的开发工具，读者可以在其中选择一种来开发 Windows 环境中的应用程序。包含于 Microsoft Win32 Software Development Kit(SDK)中的文档就是专门介绍特定 API 的一种信息资源。

怎样应用指南信息

本书鼓励读者在 Windows 应用程序设计过程中采用可视化、功能性的用户界面。尽管如此，读者仍然按照自己的意愿，并且根据自己软件的特点对指南信息进行自由的取舍。但是，通过理解本书的这些指南信息，读者就可以让自己的用户把他们的工作技能和经验从一个任务传送到另外一个任务，并且使他们对新任务的学习变得更加容易了。除此之外，随着程序设计朝着以数据为中心发展，传统应用程序领域内的一些界限也正在慢慢地被打破。这样一来，程序界面就变得越来越不一致；对于用户来说这种情况让他们感到越来越迷惑了。

相反地，对设计指南盲目地追随并不能保证最终程序的可用性。本书是程序设计人员相当有价值的一个工具，但是这个工具必须和其他的因素组合起来才能发挥它应有的作用，从而设计出一个有效的软件。在这里，“其他的因素”包括设计原理的应用、任务分析、设计原型以及可用性评定等等。

假若读者在本书内容的原理基础上进行了自己的发展，并且维持了 Windows 界面可视化和用户行为规范的一个可靠的级别，那么就可以说读者已经真正读懂了本书。在通常的情况下，除非界面对新的元素或者用户行为提供了支持，否则请不要轻率地添加新的界面元素。对于一套具体的软件来说，相关的用户对这个软件的界面工作情况已经有了一个先入为主的期望。所以说，如果出现了期望和实际不一致的情况，那么这不仅仅会让用户感到迷惑，而且还会增加不必要的复杂性。

本书为读者提供的指南信息完全可以取代其他针对 Windows 3.1 以及所有 Windows 早期版本而出版的指南书籍。本书特别适于用作 Microsoft Windows，Microsoft Windows NT

Workstation 以及 Microsoft Windows NT Server 环境下的应用程序开发指南。本书和针对其他操作系统而编写的指南之间没有任何直接的联系。

如果希望了解在 Windows 95 和 Windows NT 操作系统环境中开发应用程序的情况，请读者参阅附录 D“Windows 特殊版本的支持”。

目 录

第二部分 Windows界面组件

第三部分 设计规范和准则

附 录

第一部分
用户交互设计基础

第一章　设计原理和方法

对于一个具有良好用户界面设计的应用程序来说，它的界面设计是建立在正确的设计原理和正确的开发过程基础上的，整个设计过程都应该是以用户以及他们的任务为中心来展开的。本章总结了 Microsoft Windows 应用程序界面设计的基本原理，同时也介绍了一个有效的人机界面设计过程中需要用到的技术以及方法。

1.1　以用户为中心的设计原理

这一部分的内容向读者描述了 Windows 和本书内容引以为基础的设计原理。在设计 Windows 环境下的应用程序的时候，读者就会发现这些原理是非常有价值的。

1.1.1　用户控制

用户界面设计中一个重要的原理就是用户应该时时刻刻感到是自己在对软件进行控制，而不是软件在控制着用户。在这个原理的后面隐藏着大量的含义。

第一个含义就是由用户来指定操作方式，而不是由计算机或者软件来指定操作方式。在这里，用户扮演的应该是一个主动的角色，而不是一个被动的角色。用户可以利用许多技巧来自动地完成一项具体的任务，但是同时必须允许用户对这个自动过程能够进行选择或者控制。

第二个含义是设计人员的软件应该尽可能地具有交互作用的能力并且能够对用户的操作尽可能敏感地作出反应。在软件设计的过程中，设计人员应该尽可能地避免使用“模式”。在这里，“模式”的含义是指排斥了常规交互作用或者限制用户进行特定交互作用的一种情况。如果模式是唯一或者最好的一种选择（例如，在一个绘图程序中选择一个特定的工具），那么请确保这个模式的选择是明显的、可视的，并且显而易见是用户选择的结果，并且可以

很容易地退出这个选择。

如果想要了解用户控制设计原理具体应用的情况，请读者参阅第四章“输入基础”和第五章“常规交互技术”中的内容。这两章介绍了软件应该支持的所有基本的交互作用形式。

1.1.2　直接性

设计人员设计的软件应该可以让用户直接处理以软件形式表达的信息。无论是拖动一个对象对其重新进行定位，还是在一个文档中漫游到一个新的位置，用户都应该在屏幕中看到自己的操作对对象产生的影响。信息和选择的可视性同样也可以减少用户大脑的工作量。和回忆起一条命令的语法规则比较起来，用户识别出一条命令要容易得多。

熟悉的隐喻可以为用户的任务提供一个直接和直观的的界面。如果允许用户对他们的知识和经验进行传达，那么隐喻就可以帮助用户更加容易地预知和学习基于软件的表达方式。

在使用隐喻的时候，读者没有必要把一个基于计算机的应用限制在它的“真实世界”应用中。举个例子来说，和基于纸张的应用不一样，Windows 桌面中的一个文件夹可以用于组织大量的对象，比如打印机、计算器以及其他的更小的文件夹等等；它已经不再是“真实世界”中文件夹的原意了。类似地，一个 Windows 文件夹可以更加容易地进行排序处理。在界面设计中使用隐喻的目的是为了向大家提供一个易于理解的形式：隐喻代表的并不仅仅是它本身的含义！

隐喻支持一个人的认知能力，而不是他（她）的记忆力。和回忆起一个特定命令的名称比较起来，人们能够更加容易地记起与一个熟悉对象有关联的含义。

如果想要了解直接性和隐喻原理具体应用的情况，请读者参阅第五章“常规交互技术”和第十三章“可视化设计”中的内容。这两个章节各自介绍了界面中直接性的应用情况（包括“拖动和放置”）以及图标和其他图形化元素设计过程中使用隐喻的情况。

1.1.3　一致性

一致性允许用户把现成的知识传递到新的任务中，这样就可以更加快速地学习到新的知识。由于用户不需要花时间去记住交互作用中的不同之处，所以就可以把自己的主要精力集中于具体的任务上面去。由于提供了稳定、连续的操作方式，所以一致性可以让不同任务中的界面为用户带来熟悉的感觉，并且所有的操作都是可以预见的，用户不用学习新的操作方式。

在所有界面元素的设计过程中，保持一致性都是最重要的；在这里，界面元素包括命令名称、可视化的信息显示以及用户的操作行为。为了在软件的设计中体现出一致性，用户必须认识到下面几个方面的问题：

- 在一个产品中的一致性。利用一致的命令和界面集提供常规的功能。举个例子来说，设计人员需要使用 Copy 命令的时候需要谨慎，它在某种情况下面表示立即执行一个操作，但是在另外一个情况下面则表示一个对话框，用户需要在这个对话框中输入一个文件拷贝的目标位置。通过这个例子，我们可以看出：需要利用用户看上去差不多的命令来执行某种特定的功能。
- 在操作环境中的一致性。通过在交互作用和 Windows 提供的界面规范之间维持一个

高等级的一致性，设计人员所设计的软件就可以从用户已经学到的、交互技巧的应用能力中获益。

- 隐喻的一致性。和隐喻的一个对象含义比较起来，如果某种特定的操作行为显得更加富有特色，那么用户有可能就会难以把那种行为与一个对象联系起来。举个例子来说，假设我们现在需要一个隐喻的东西来代表其中所放置对象的可恢复能力。如果我们选用的是一个焚化炉，那么和一个废纸篓比较起来，显然前者已经偏离了我们本来的意图。

虽然一致性原理的应用是本书需要说明的基本问题，但是下述章节集中讲述的元素对于所有基于 Windows 的软件来说都是适用的：第六章“窗口”、第七章“菜单、控件和工具栏”和第八章“二级窗口”。如果希望知道自己的软件怎样才能与 Windows 紧密地集成，请读者参阅第十章“系统集成”和第十一章“OLE 嵌入和 OLE 链接对象的处理”中的内容。

1.1.4 容错

用户都喜欢对软件的操作界面进行探究，而且总是在试验和出错的情况下面学到操作知识的。一个设计有效的界面允许用户进行交互式的探究，它只为用户提供了恰当的选择集合，当用户的探究活动有可能破坏系统或者数据，或者将要出现稍好的一种情况，那就是进行可恢复的操作的时候，它就会向用户指出潜在的危险，并且发出相应的警告。

即使在设计得最好的一个操作界面中，用户都有可能会犯下错误。这些错误既有可能是物理性的（不小心指向了错误的命令或者数据），也有可能是主观性的（对于相关命令或者数据的选择作出了错误的决定）。一个设计有效的界面可以防止可能导致错误的情况产生；它也可以调整潜在的用户错误，并且让用户通过简便的方式取消自己错误的操作。

如果希望了解“容错”原理在具体设计过程中如何应用的情况，请读者参阅第十二章“用户帮助”中的内容。在这一章中，我们为读者提供了界面设计中与可恢复性支持有关的信息，其中涉及到了上下文相关的用法、任务关联以及用户帮助的参考形式等等。如果希望了解面向大范围内用户如何进行程序设计的信息，请读者参阅第十四章“特殊的设计考虑”中介绍的内容。

1.1.5 反馈

在程序设计的过程中，设计人员时刻都需要注意为用户的操作提供恰当的反馈信息。这些反馈信息必须是可视化的，有时候还有可能是一些声音信息。针对用户的每一个交互式操作，程序都需要为这个操作提供相应的信息，从而确定该软件正在对用户的输入作出响应，并且体现区分出操作本质不同的一些细节信息。

有效的反馈是非常及时的，并且要尽可能地接近用户的交互操作点。即使计算机正在处理一项特殊的任务，它都应该为用户提供关于处理情况的信息，并且有可能的前提下还要指出通过怎样的操作才能退出这个处理过程。一个对于任何输入都不作响应的、死气沉沉的屏幕最能让用户感到不知所措。对于一个典型的用户来说，他（她）能够容忍的、屏幕不作任何响应的时间只有几秒钟。

设计人员提供的操作反馈信息类型必须和具体的任务相对应，这一点同样也是相当重要的。鼠标指示符的改变或者一条状态条消息可以表达一些简单的信息；更加复杂的反馈信

息需要设计人员使用一个消息框来表达。

如果希望了解怎样具体应用可视化和声音反馈等信息，请读者参阅第十三章“可视化设计”和第十四章“特殊的设计考虑”中的内容。

1.1.6　美学效果

可视化设计是软件界面设计中一个非常重要的组成部分。可视化属性能够为用户带来深刻的印象，并且为特定对象交互行为的表达提供了重要的线索，这样就使得程序界面变得更加容易理解和操作了。同时，设计人员需要记住的重要一点在于：在屏幕中显示出来的每一个可视化元素都潜在地分散用户的注意力。提供一个舒适的操作环境有助于帮助用户对显示信息的理解。如果需要达到程序设计的美学效果，一个图形或者视觉设计人员是必不可少的。

如果希望了解和界面美学效果有关的信息和指南，请读者参阅第十三章“可视化设计”。在这一章中，我们将全方位地讨论从独立元素设计到字模用法以及窗口布局的问题。

1.1.7　简化性

一个界面应该是比较简单的(不是过分的简单化)，用户易于对其进行学习和使用。这个界面也必须为用户提供对应用程序所具有的全部功能的访问权利。在界面的设计过程中，把功能发挥到最大，同时还要保持操作的简便性，这两个方面是互相矛盾的。对于一个有效的界面设计来说，它总是能够保持这两个方面之间的平衡。

支持简化性的一种途径是减少辅助信息的介绍，使用最少的信息把问题说明白。举个例子来说，对于命令名称或者消息来说，应该防止使用多余的描述。不恰当的或者冗长的惯用语将搞乱自己的设计；对于用户来说，他们也不能正确地找出基本的信息。如果想要设计出一个虽然简单，但是有用的界面，另外一条途径就是使用自然的语言，不要故意做作。元素本身的安排和陈述对于用户正确理解它们的意义和组织来说是十分重要的。

用户也可以采用“逐步展示”(progressive disclosure)手段来帮助用户理解界面操作的复杂性。在这里，“逐步展示”需要设计人员对信息进行精心的组织，并且只有在合适的时间里面才让它们显示出来。通过把提供给用户的信息“隐藏”起来，设计人员就可以减少对信息的处理量。举个例子来说，如果用户单击一个菜单，那么就会显示出其中的选项；对话框的使用可以减少菜单选项的数量。

“逐步展示”并不意味着设计人员需要利用非传统的技术才能向用户透露信息(比如需要按下一个特定的键才能访问基本的功能或者强迫用户按照一个更长的阶层式交互序列进行操作等等)，因为这样做就会使界面的设计变得更加复杂和麻烦了。

如果希望了解有关简化性原理应用更多的信息，请读者参阅第七章“菜单、控件和工具栏”。在第七章，我们将和读者一起详细地讨论“逐步展示”的问题，并且介绍在自己设计的界面中怎样和在何时使用标准(由系统提供)的元素。

1.2　设计方法

有效的界面设计并不是只是按照一系列的规则进行操作就可以实现了，它需要设计人

员具有以用户为中心的设计态度，并且采用正确的设计方法才能实现。有效的界面设计也涉及到早期对界面进行的规划以及整个软件开发过程中进行的工作情况。

1.2.1　一个配备平衡的设计队伍

软件产品设计中一个重要的考虑就是它的设计和实施队伍的组成。一般说来，我们首先都需要对软件产品的开发、可视化设计、正文写作、人文因素以及可用性评定人员力量进行平衡。对于一个独立的人员来说，在他身上很难找到所有的这些素质。因此，我们需要建立在这些领域内各有所长的人员队伍，这样有助于开发出一个比较完美的最终产品。

请确保这一支设计队伍可以有效地工作，并且互相之间能够顺利地进行交流。把他们的工作地点安排得比较近，或者都安排到一个地方里面，这样才能让他们就设计过程中的某些细节更好地进行交流、切磋。

1.2.2　设计周期

对于一个有效的、以用户为中心的软件开发过程来说，我们可以把这个过程分成几个重要的阶段来完成：设计、原型化、测试以及反复。下面这几个部分为读者详细介绍了这些阶段。

1. 设计

一个软件开发过程中最初的工作是最为关键的，而设计就是最为关键的工作。为什么说它是最关键的呢？这是由于在这个阶段中，设计人员需要从总体上决定自己产品的轮廓是什么样子。如果在基础工作上面没有花力气，从而存在着漏洞，那么在以后的阶段中就很难进行纠正了。

在“设计”这个开发阶段中，不仅仅要涉及到软件产品外观和特性的定义，而且还需要设计人员理解这个产品的用户是谁以及他们的任务、想法以及最终的目标是什么。除此之外，该阶段还需要设计人员对其他元素具有一定的理解，比如对用户背景（年龄、性别、专业知识、经验水平、身体缺陷以及其他一些特殊的需要）、工作环境（设备配置情况、社会和文化环境以及物质环境）以及用户当前任务组织（需要的步骤、基础、多余的活动以及输出情况）等等都需要具有清楚的认识。和其他的信息集散地比较起来，一个设计良好的软件系统所拥有的用户和用户各自的要求都是大为不同的。

在这个阶段中，设计人员需要从概念上定义好自己产品的框架，它从一定程度上要考虑到该软件最终用户的知识和经验水平。在理想的情况下面，设计人员应该创建出一个设计模型，这个模型应该反映出最终用户工作任务的具体要求。设计人员也应该选择好基本的组织方式以及不同类型的隐喻。如果希望最后使用的是一些有效的隐喻，那么对最终用户所完成任务的全面认识将是非常重要的。

设计人员还需要写好一个软件产品的说明文档。设计一个好的写作格式不仅仅提供了一个有价值的参考资源和沟通形式，而且还可以帮助用户更加彻底、更加集中地解决碰到的一些实际问题。

2. 原型化

当设计人员定义好一个设计模型以后，就可以着手对设计的基本部分进行原型化处理了。这个过程可以采用“纸笔”模式来完成，设计人员可以创建自己想象中界面的图样，其中可以附上其他的元素：故事板(storyboards)——类似于连环画的草图系列，它图解了特定的处理过程；动画——类似于电影的操作模拟；或者利用原型化工具或者由普通开发工具设计出来的一个软件，设计人员可以在其中进行操作模拟。

从几个方面来说，原型都就是一种非常有价值的东西。首先，它为设计过程中的沟通提供了一个有效的工具。其次，它可以帮助设计人员定义工作流程，并且能够更好地把设计过程形象化。第三，它以最低的代价模拟了一个设计过程中用户的实际情况。在软件开发的初期，这一点是相当有用的。

设计人员建立的原型类型将取决于自己的具体设计目标。功能、任务流程、界面、操作以及文档都只是一个软件产品中需要进行访问的一些不同的方面。举个例子来说，在定义任务组织或者概念上的想法的时候，设计人员就需要用到“纸笔”模式或者故事板。对于用户交互操作的技巧来说，可以运行的原型通常是最好的。

设计人员还需要考虑是否把原型的深度和宽度变得更大一些。原型的宽度越大，就需要试着在其中包含更多的特性，这样才能帮助理解最终用户对于概念和组织会作出什么样的反应。如果设计人员把自己的精力更多地放在设计区域或者一个具体特性的细节性用法上面，那么就需要使原型的深度变得更大，从而在其中包含针对某个给定特性或者任务更多的细节。

3. 测试

以用户为中心的测试需要涉及到设计过程中的用户。对于一个设计或者设计某个方面的可用性测试为程序设计人员提供了相当有价值的信息，它是一个软件产品开发成功的前提。质量保证测试与可用性测试具有很多的不同之处。质量保证测试的目的是发现编程中出现的错误，而可用性测试则是为了了解程序界面在多大程序上能够满足用户的需求和期望。当然，编程上的错误有时候也有可能会影响到用户对界面的操作。

测试能够达到的目的有许多种。我们可以利用测试来发现初步设计中潜在的问题。我们也可以把自己的注意力放在两个或者多个设计方案的比较性研究上面，从而决定在给定一个特定任务或者一个任务集合的前提下面，哪一种设计方案更好。

可用性测试不仅仅可以让程序设计人员更加有效地完成开发任务，而且还为他们提供了大量的实测数据，这些数据可以帮助他们对自己的程序作出有效的修改。可用性测试还提供了与用户的感觉、满意程度、疑问以及问题有关的信息，设计人员对这些信息的掌握是相当必要的。

在进行测试的过程中，设计人员要注意自己选择的合作人员必须符合自己最终用户的种种条件。在自己的部门中找到一个同事，要求他来帮助自己完成软件的测试是相当容易的，但是软件的最终用户很少有人具有与程序设计人员相同的经验和水平。在下面的一个部分“设计过程中的可用性评定”中，我们向读者介绍了与可用性测试管理有关的信息。