

GUIA DE PREGUNTAS

Material "DISEÑO DE INTERFAZ DE USUARIO.
Ingeniería del Software (Capítulo 15) de Roger S. Pressman (5^{ta} Edición)"

1. Enuncie las áreas de interés en las que se centra el diseño de interfaz.
2. Enuncie las tres reglas que propone Theo Mantel para el diseño de interfaces.
3. Enuncie los principios de diseño propuestos por Mandel que permiten dar el control al usuario.
4. Enuncie los principios de diseño propuestos por Mandel que permiten reducir la carga de memoria del usuario.
5. Defina interfaz que en forma consecuente adquiera y presente la información.
6. Enuncie los principios de diseño propuestos por Mandel para ayudar a construir una interfaz consecuente.
7. Enuncie los modelos que entran en juego al diseñar la interfaz de usuario.
8. Enuncie categorías de usuarios.
9. Defina proceso de diseño de la interfaz, enuncie etapas y formule un esquema.
10. Enuncie preguntas que se formulan para el análisis del entorno de usuario.
11. Enuncie en que se centra la validación de la interfaz de usuario.
12. Enuncie pasos de diseño de interfaz.
13. Enuncie temas de diseño que deben abordarse cuando se va a considerar un servicio de ayuda.
14. Enuncie características de los mensajes de error.
15. Enuncie temas de diseño que surgen cuando el modo de interacción son órdenes escritas.
16. Defina sistemas de desarrollo de la interfaz de usuario (SDIU).
17. Enuncie mecanismos que proporciona un sistema de desarrollo de la interfaz de usuario.
18. Enuncie criterios de evaluación del modelo de diseño de interfaz y de un esquema de su ciclo de evaluación.

EL plano de una casa (su diseño arquitectónico) no está completo sin la representación de puertas, ventanas y conexiones de servicios para el agua, electricidad y teléfono (por no mencionar la televisión por cable). Las «puertas, ventanas y conexiones de servicios» del software informático es lo que constituye el diseño de la interfaz de usuario.

El diseño de la interfaz se centra en tres áreas de interés: (1) el diseño de la interfaz entre los componentes del software; (2) el diseño de las interfaces entre el software y los otros productores y consumidores de información no humanos (esto es, otras entidades externas) y (3) el diseño de la interfaz entre el hombre (esto es, el usuario) y la computadora. En este capítulo nos centraremos exclusivamente en la tercera categoría de diseño de interfaz —el diseño de la interfaz de usuario—.

Ben Shneiderman [SHN87] habla sobre esta categoría de diseño en el prólogo de su libro y afirma lo siguiente:

Para muchos usuarios de sistemas de información computerizados la frustración y la ansiedad forman parte de su vida diaria. Luchan por aprender el lenguaje de órdenes y los sistemas de selección de menús que supuestamente les ayudan a realizar su trabajo. Algunas personas se encuentran con casos tan serios de shocks informáticos, terror en el terminal o neurosis en la red, que evitan utilizar sistemas computerizados.

VISTAZO RÁPIDO

¿Qué es? El diseño de la interfaz de usuario es la categoría de diseño que crea un medio de comunicación entre el hombre y la máquina. Con un conjunto de principios para el diseño de la interfaz, el diseño identifica los objetos y acciones de la interfaz y crea entonces un formato de pantalla que formará la base del prototipo de interfaz de usuario.

¿Quién lo hace? El ingeniero del software es quien diseña la interfaz de usuario mediante la aplicación del proceso iterativo que se sirve de los principios predefinidos del diseño.

¿Por qué es importante? Si el software es difícil de utilizar, si obliga a cometer errores, o causa frustración para conseguir los objetivos, no será de agrado,

independientemente de la potencia informática que demuestre o de la funcionalidad que ofrezca. Dado que la interfaz es la que da forma a la percepción del software por parte del usuario, tiene que estar bien diseñada.

¿Cuáles son los pasos? El diseño de la interfaz de usuario comienza con la identificación de los requisitos del usuario, de la tarea y del entorno. Una vez identificadas las tareas, se crean y se analizan los escenarios del usuario para definir el conjunto de objetos y de acciones de la interfaz. Esto es lo que forma la base para la creación del formato de la pantalla que representa el diseño gráfico y la colocación de iconos, la definición del texto descriptivo en pantalla, la

especificación y títulos de las ventanas, y la especificación de los elementos principales y secundarios del menú. Las herramientas se utilizan para generar prototipos y por último implementar el modelo de diseño y evaluar la calidad del resultado.

¿Cuál es el producto obtenido? La creación de escenarios de usuarios y la generación de formatos de pantallas. Y el desarrollo y modificación iterativo de prototipos.

¿Cómo puede estar seguro de que lo ha hecho correctamente? Los usuarios controlan el prototipo mediante pruebas y la respuesta obtenida del control del texto se utiliza para la siguiente modificación iterativa del prototipo.

Los problemas a los que alude Shneiderman son reales. Es cierto que las interfaces gráficas, ventanas, iconos y selecciones mediante ratón han eliminado muchos de los terribles problemas con la interfaz. Pero incluso en un «mundo de ventanas» todos encontramos interfaces de usuario difíciles de aprender, difíciles de utilizar, confusas, imperdonables y en muchos casos totalmente frustrantes. Sin embargo, hay quien dedica tiempo y energías construyendo estas interfaces, y es posible que estos problemas no los crearan a propósito.

15.1 LAS REGLAS DE ORO

Theo Mantel [MAN97] en su libro crea tres «reglas de oro» para el diseño de la interfaz:

1. Dar el control al usuario
2. Reducir la carga de memoria del usuario
3. Construir una interfaz consecuente

Estas reglas de oro forman en realidad la base para los principios del diseño de la interfaz de usuario que servirán de guía para esta actividad de diseño de software tan importante.

15.1.1. Dar el control al usuario

Durante la sesión de recopilación de los requisitos para un nuevo sistema de información, un usuario clave fue preguntado a cerca de los atributos de la interfaz gráfica orientada a ventanas.

El usuario respondió solemnemente, «Lo que me gustaría realmente es un sistema que lea mi mente. Que conozca lo que quiero hacer antes de necesitarlo y que me facilite hacerlo. Eso es todo. Simplemente eso.»

Mi primera reacción fue mover la cabeza y sonreír, y hacer una pausa por unos instantes. No había nada malo en la solicitud del usuario. Lo que quería era que un sistema reaccionara ante sus necesidades y que le ayudara a hacer las cosas. Quería controlar la computadora, y no dejar que la computadora le controlara.

La mayor parte de las restricciones y limitaciones impuestas por el diseñador se han pensado para simplificar el modo de interacción. Pero, ¿para quienes? En muchos casos es posible que el diseñador introduzca limitaciones y restricciones para simplificar la implementación de la interfaz. Y el resultado puede ser una interfaz fácil de construir, pero frustrante de utilizar.

Mandel [MAN97] define una serie de principios de diseño que permiten dar control al usuario:

Definir los modos de interacción de manera que no obligue a que el usuario realice acciones innecesarias y no deseadas. Un modo de interacción es el estado actual de la interfaz. Por ejemplo, si en el procesador de textos se selecciona el *corrector ortográfico*, el software pasa a modo corrector ortográfico. No hay ninguna razón por la que obligar a que el usuario permanezca en este modo si el usuario desea continuar editando una parte pequeña de texto. El usuario deberá tener la posibilidad de entrar y salir de este modo sin mucho o ningún esfuerzo.



¿Cómo se diseñan interfaces que den el control al usuario?

Tener en consideración una interacción flexible. Dado que diferentes usuarios tienen preferencias de interacción diferentes, se deberán proporcionar diferentes selecciones. Por ejemplo, un software que pueda permitir al usuario

interactuar a través de las órdenes del teclado, con el movimiento del ratón, con un lápiz digitalizador, mediante órdenes para el reconocimiento de voz. Sin embargo, no toda acción responde a todo mecanismo de interacción. Considere por ejemplo, la dificultad de utilizar órdenes del teclado (o entrada de voz) para dibujar una forma compleja.

Permitir que la interacción del usuario se pueda interrumpir y deshacer. Cuando un usuario se ve involucrado en una secuencia de acciones, deberá poder interrumpir la secuencia para hacer cualquier otra cosa (sin perder el trabajo que se hubiera hecho anteriormente). El usuario deberá también tener la posibilidad de «deshacer» cualquier acción.

Aligerar la interacción a medida que avanza el nivel de conocimiento y permitir personalizar la interacción. El usuario a menudo se encuentra haciendo la misma secuencia de interacciones de manera repetida. Merece la pena señalar un mecanismo de «macros» que posibilite al usuario personalizar la interfaz y así facilitar la interacción.

Cita:

Uno de los errores más comunes que se comete cuando intentamos diseñar algo a prueba de tontos es subestimar la ingenuidad de los tontos.
Douglas Adams

Ocultar al usuario ocasional los entresijos técnicos. La interfaz de usuario deberá introducir al usuario en el mundo virtual de la aplicación. El usuario no tiene que conocer el sistema operativo, las funciones de gestión de archivos, o cualquier otro secreto de la tecnología informática. Esencialmente, la interfaz no deberá requerir nunca que el usuario interactúe a un nivel «interno» de la máquina (por ejemplo, el usuario no tendrá que teclear nunca las órdenes del sistema operativo desde dentro del software de aplicación).

Diseñar la interacción directa con los objetos que aparecen en la pantalla. El usuario tiene un sentimiento de control cuando manipula los objetos necesarios para llevar a cabo una tarea de forma similar a lo que ocurriría si el objeto fuera algo físico. Como ejemplo de manipulación directa puede ser una interfaz de aplicación que permita al usuario «alargar» un objeto (cambiar su tamaño).

15.1.2. Reducir la carga de memoria del usuario


Cuanto más tenga que recordar un usuario, más propensa a errores será su interacción con el sistema. Esta es la razón por la que una interfaz de usuario bien diseñada no pondrá a prueba la memoria del usuario. Siempre que sea posible, el sistema deberá «recordar» la información per-

tinente y ayudar a que el usuario recuerde mediante un escenario de interacción. Mandel [MAN97] define los principios de diseño que hacen posible que una interfaz reduzca la carga de memoria del usuario:

Reducir la demanda de memoria a corto plazo. Cuando los usuarios se ven involucrados en tareas complejas, exigir una memoria a corto plazo puede ser significativo. La interfaz se deberá diseñar para reducir los requisitos y recordar acciones y resultados anteriores. Esto se puede llevar a cabo mediante claves visuales que posibiliten al usuario reconocer acciones anteriores sin tenerlas que recordar.

Establecer valores por defecto útiles. El conjunto inicial de valores por defecto tendrá que ser de utilidad para al usuario, pero un usuario también deberá tener la capacidad de especificar sus propias preferencias. Sin embargo, deberá disponer de una opción de «reinicialización» que le permita volver a definir los valores por defecto.

Definir las deficiencias que sean intuitivas. Cuando para diseñar un sistema se utiliza la mnemónica (por ejemplo, alt-P para invocar la función de imprimir), ésta deberá ir unida a una acción que sea fácil de recordar (por ejemplo, la primera letra de la tarea que se invoca).

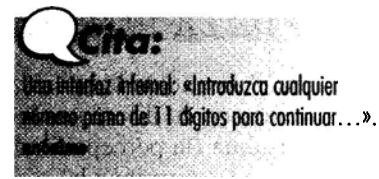
 ¿Cómo se pueden diseñar interfaces que reduzcan la carga de memoria del usuario?

El formato visual de la interfaz se deberá basar en una metáfora del mundo real. Por ejemplo, en un sistema de pago de facturas se deberá utilizar la metáfora de la chequera y el registro del cheque para conducir al usuario por el proceso del pago de facturas. Esto hace posible que el usuario comprenda bien las pistas y que no tenga que memorizar una secuencia secreta de interacciones.

Desglosar la información de forma progresiva. La interfaz deberá organizarse de forma jerárquica. Esto es, en cualquier información sobre una tarea se deberá presentar un objeto o algún comportamiento en primer lugar a un nivel alto de abstracción. Y solo después de que el usuario indique su preferencia realizando la selección mediante el ratón se presentarán más detalles. Un ejemplo muy común en muchas aplicaciones de procesamiento de texto es la función de subrayado dado que es una función que pertenece al menú de estilo de texto. Sin embargo no se muestran todas las posibilidades de subrayado. El usuario es el que debe seleccionar el subrayado, y así se presentarán entonces las opciones de esta función (por ejemplo, subrayado sencillo, subrayado doble, subrayado de guiones).


15.1.3. Construcción de una interfaz consistente

La interfaz deberá adquirir y presentar la información de forma consecuente. Esto implica (1) que toda la información visual esté organizada de acuerdo con el



diseño estándar que se mantiene en todas las presentaciones de pantallas; (2) que todos los mecanismos de entrada se limiten a un conjunto limitado y que se utilicen consecuentemente por toda la aplicación, y que (3) los mecanismos para ir de tarea a tarea se hayan definido e implementado consecuentemente. Mandel [MAN97] define un conjunto de principios de diseño que ayudan a construir una interfaz consecuente:

Permitir que el usuario realice una tarea en el contexto adecuado. Muchas interfaces implementan capas complejas de interacciones con docenas de imágenes de pantallas. Es importante proporcionar indicadores (por ejemplo, títulos de ventanas, iconos gráficos, codificaciones en colores consecuentes) que posibiliten al usuario conocer el contexto del trabajo que está llevando a cabo. Además, el usuario deberá ser capaz de determinar de dónde procede y qué alternativas existen para la transición a una tarea nueva.

 ¿Cómo se pueden construir interfaces consecuentes?

Mantener la consistencia en toda la familia de aplicaciones. Un conjunto de aplicaciones (o productos) deberá implementar las mismas reglas de diseño para mantener la consistencia en toda la interacción.

Los modelos interactivos anteriores han esperado al usuario, no realicemos cambios a menos que exista alguna razón convincente para hacerlo. Una vez que una secuencia interactiva se ha convertido en un estándar hecho (por ejemplo, la utilización de alt-S para grabar un archivo), el usuario espera utilizar esta combinación en todas las aplicaciones que se encuentre. Un cambio podría originar confusión (por ejemplo, la utilización de alt-S para invocar la función cambiar de tamaño).

Los principios del diseño de interfaces tratados aquí y en sesiones anteriores proporcionan una guía básica para la ingeniería del software. En la siguiente sección examinaremos el proceso de diseño de la interfaz.



Líneas Generales para el diseño de las interfaces.

El proceso global para el diseño de la interfaz de usuario comienza con la creación de diferentes modelos de funcionamiento del sistema (la percepción desde fuera). Es entonces cuando se determinan las tareas orientadas al hombre y a la máquina que se requieren para lograr el funcionamiento del sistema; se tienen en consideración los temas de diseño que se aplican a todos los diseños de interfaces; se utilizan herramientas para generar prototipos y por último para implementar el modelo de diseño, y evaluar la calidad del resultado.

15.2.1. Modelos de diseño de la interfaz

Cuando se va a diseñar la interfaz de usuario entran en juego cuatro modelos diferentes. El ingeniero del software crea un *modelo de diseño*; cualquier otro ingeniero (o el mismo ingeniero del software) establece un *modelo de usuario*, el usuario final desarrolla una imagen mental que se suele llamar *modelo de usuario o percepción del usuario*, y los que implementan el sistema crean una *imagen de sistema* [RUBSS]. Desgraciadamente, todos y cada uno de los modelos pueden diferir significativamente. El papel del diseñador de interfaz es reconciliar estas diferencias y derivar una representación consecuente de la interfaz.



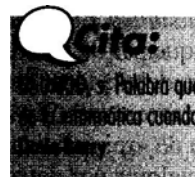
Una fuente excelente de directrices de diseño y referencias se puede encontrar en www.ibm.com/ibm/easy/

Un modelo de diseño de un sistema completo incorpora las representaciones del software en función de los datos, arquitectura, interfaz y procedimiento. La especificación de los requisitos puede que establezca ciertas limitaciones que ayudarán a definir al usuario del sistema, pero el diseño de la interfaz suele ser un único tema secundario de modelo de interfaz¹.

El modelo de usuario representa el perfil de los usuarios finales del sistema. Para construir una interfaz de usuario efectiva, «todo diseño deberá comenzar por conocer los usuarios destino, así como los perfiles de edad, sexo, habilidades físicas, educación, antecedentes culturales o étnicos, motivación, objetivos y personalidad» [SCH87] Además de esto se pueden establecer las siguientes categorías de usuarios:

- **principiantes:** en general no tienen *conocimientos sintácticos*² ni *conocimientos semánticos*³ de la utilización de la aplicación o del sistema;

- **usuarios esporádicos y con conocimientos:** poseen un conocimiento semántico razonable, pero una retención baja de la información necesaria para utilizar la interfaz;
- **usuarios frecuentes y con conocimientos:** poseen el conocimiento sintáctico y semántico suficiente como para llegar al «síndrome del usuario avanzado»), esto es, individuos que buscan interrupciones breves y modos abreviados de interacción.



La percepción del sistema (el modelo de usuario) es la imagen del sistema que el usuario final tiene en su mente. Por ejemplo, si se preguntara a un usuario de un procesador de texto en particular que describiera su forma de manejar el programa, la respuesta vendría guiada por la percepción del sistema. La precisión de la descripción dependerá del perfil del usuario (por ejemplo, los principiantes harían lo posible por responder con una respuesta muy elemental) y de la familiaridad global con el software del dominio de la aplicación. Un usuario que comprenda por completo los procesadores de texto, aunque puede que haya trabajado solo una vez con ese procesador específico, es posible que proporcione de verdad una descripción más completa de su funcionamiento que el principiante que haya pasado unas cuantas semanas intentando aprender el funcionamiento del sistema.

La imagen del sistema es una combinación de fachada externa del sistema basado en computadora (la apariencia del sistema) y la información de soporte (libros, manuales, cintas de vídeo, archivos de ayuda) todo lo cual ayuda a describir la sintaxis y la semántica del sistema. Cuando la imagen y la percepción del sistema coinciden, los usuarios generalmente se sienten a gusto con el software y con su funcionamiento. Para llevar a cabo esta «mezcla» de modelos, el modelo de diseño deberá desarrollarse con el fin de acoplar la información del modelo de usuario, y la imagen del sistema deberá reflejar de forma precisa la información sintáctica y semántica de la interfaz.

Los modelos descritos anteriormente en esta sección son «abstracciones de lo que el usuario está haciendo o piensa que está haciendo o de lo que cualquier otra per-

¹ Por supuesto esto no es como debena ser. Para sistemas interactivos, el diseño de la interfaz es tan importante como el diseño de los datos, arquitectura o el de componentes.

² En este contexto el conocimiento sintáctico se refiere a la mecánica de interacción que se requiere para utilizar la interfaz de forma eficaz.

³ El conocimiento semántico se refiere al sentido subsiguiente de aplicación —una comprensión de la realización de todas las funciones, del significado de entrada y salida, de las metas y objetivos del sistema—.

sona piensa que debería estar haciendo cuando utiliza un sistema interactivo» [MON84]. Esencialmente, estos modelos permiten que el diseñador de la interfaz satisfaga un elemento clave del principio más importante del diseño de la interfaz de usuario: «quien conoce al usuario, conoce las tareas».



Cuando la imagen del sistema y la percepción del sistema coinciden, el usuario puede utilizar la aplicación de forma efectiva.

15.2.2. El proceso de diseño de la interfaz de usuario

El proceso de diseño de las interfaces de usuario es iterativo y se puede representar mediante un modelo espiral similar al abordado en el Capítulo 2. En la Figura 15.1 se puede observar que el proceso de diseño de la interfaz de usuario acompaña cuatro actividades distintas de marco de trabajo [MAN97]:

1. Análisis y modelado de usuarios, tareas y entornos.
2. Diseño de la interfaz
3. Implementación de la interfaz
4. Validación de la interfaz

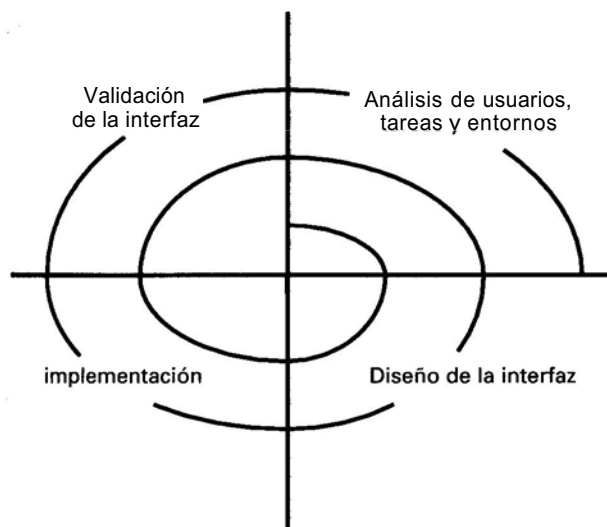
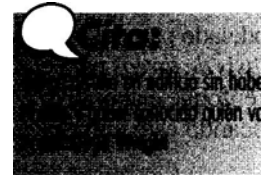


FIGURA 15.1. El proceso de diseño de la interfaz de usuario.

La espiral que se muestra en la Figura 15.1 implica que cada una de las tareas anteriores aparecerán más de una vez, en donde a medida que se avanza por la espiral se representará la elaboración adicional de los requisitos y el diseño resultante. En la mayoría de los casos, la actividad de implementación implica la generación de prototipos —la única forma práctica para validar lo que se ha diseñado—.

La actividad de análisis inicial se concentra en el perfil de los usuarios que van a interactuar con el sistema.

Se registran el nivel de conocimiento, la comprensión del negocio y la receptividad general del nuevo sistema, y se definen diferentes categorías de usuarios. En cada categoría se lleva a cabo la elicitación de los requisitos. Esencialmente, el ingeniero del software intenta comprender la percepción del sistema (Sección 15.2.1) para cada clase de usuario.

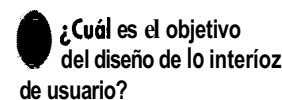


Una vez definidos los requisitos generales, se lleva a cabo un análisis más detallado de las tareas. Se identifican, describen y elaboran las tareas que lleva a cabo el usuario para conseguir los objetivos (por encima de la cantidad de pasos iterativos a través de la espiral). El análisis de tareas se estudia detalladamente en la Sección 15.3.

El análisis del entorno de usuario se centra en el entorno del trabajo físico. Entre las preguntas que se formulan se encuentran las siguientes:

- ¿Dónde se ubicará físicamente la interfaz?
- ¿Dónde se situará el usuario? ¿Llevará a cabo tareas no relacionadas con el interfaz?
- ¿Se adapta bien el hardware a las limitaciones de luz, espacio o ruido?

Esta recopilación de información que forma parte de la actividad de análisis se utiliza para crear un modelo de análisis para la interfaz. Mediante esta base de modelo se comienza la actividad de diseño.



El objetivo del diseño de la interfaz es definir un conjunto de objetos y acciones de interfaz (y sus representaciones en la pantalla) que posibiliten al usuario llevar a cabo todas las tareas definidas de forma que cumplan todos los objetivos de usabilidad definidos por el sistema. El diseño de la interfaz se aborda más detalladamente en la Sección 15.4.

La actividad de implementación comienza normalmente con la creación de un prototipo que permita evaluar los escenarios de utilización. A medida que avanza el proceso de diseño iterativo, y para completar la construcción de la interfaz, se puede utilizar un kit de herramientas de usuario (Sección 15.5).

La validación se centra en: (1) la habilidad de la interfaz para implementar correctamente todas las tareas del usuario, para acoplar todas las variaciones de tareas, y para archivar todos los requisitos generales del usuario; (2) el grado de facilidad de utilización de la interfaz y de aprendizaje, y (3) la aceptación de la interfaz por parte del usuario como una herramienta útil en su trabajo.

13.3 ANÁLISIS Y MODELADO DE TAREAS

En el Capítulo 13 se estudió la elaboración paso a paso (llamada también refinamiento paso a paso y descomposición funcional) como mecanismo para refinar las tareas de procesamiento necesarias para que el software lleve a cabo la función deseada. Más adelante en este mismo libro tendremos en consideración el análisis orientado a objetos como enfoque de modelado para los sistemas basados en computadora. El *análisis de tareas* para el diseño de la interfaz o bien utiliza un enfoque elaborativo o bien orientado a objetos, pero aplicando este enfoque a las actividades humanas.

El análisis de tareas se puede aplicar de dos maneras. Como ya hemos destacado anteriormente, un sistema interactivo basado en computadora se suele utilizar para reemplazar una actividad manual o semi-manual. Para comprender las tareas que se han de llevar a cabo para lograr el objetivo de la actividad, un ingeniero⁴ deberá entender las tareas que realizan los hombres actualmente (cuando se utiliza un enfoque manual) y hacer corresponder estas tareas con un conjunto de tareas similar (aunque no necesariamente idénticas) que se implementan en el contexto de la interfaz de usuario. De forma alternativa, el ingeniero puede estudiar la especificación existente para la solución basada en computadora y extraer un conjunto de tareas que se ajusten al modelo de usuario, al modelo de diseño y a la percepción del sistema.

Independientemente del enfoque global utilizado para el análisis de tareas, el ingeniero deberá en primer lugar definir las y clasificarlas. Ya se ha descrito anteriormente que el enfoque es una elaboración paso a paso. Por ejemplo, supongamos que una compañía pequeña quiere construir un sistema de diseño asistido por computadora explícitamente para diseñadores de interiores. Al observar a un diseñador de interiores en su trabajo, el ingeniero se da cuenta y notifica que el diseño interior se compone

de una serie de actividades importantes: diseño del mobiliario, selección de tejidos y materiales, selección de decorados en paredes y ventanas, presentación (al cliente), costes y compras. Todas y cada una de estas tareas pueden elaborarse en otras subtareas. Por ejemplo, el diseño del mobiliario puede refinarse en las tareas siguientes: (1) dibujar el plano de la casa con las dimensiones de las habitaciones; (2) ubicar ventanas y puertas en los lugares adecuados; (3) utilizar plantillas de muebles para dibujar en el plano un esbozo del mobiliario a escala; (3) mover el esbozo del mobiliario para mejorar su colocación; (5) etiquetar el esbozo del mobiliario; (6) dibujar las dimensiones para mostrar la colocación; (7) realizar un dibujo en perspectiva para el cliente. Para las otras tareas importantes del enfoque se puede utilizar un método similar.

CLAVE

Las tareas humanas se definen y se clasifican como parte del análisis de las tareas. Para refinar las tareas se utiliza un proceso de elaboración. De forma alternativa, se identifican y se refinan los objetos y las acciones.

Las subtareas (1)-(7) pueden recibir más refinamiento. Las subtareas (1)-(6) se llevarán a cabo mediante la manipulación de información y mediante la realización de acciones dentro de la interfaz de usuario. Por otro lado, la subtarea (7) se podrá llevar a cabo automáticamente en el software y dará como resultado muy poca interacción directa con el usuario. El modelo de diseño de la interfaz deberá adaptarse a cada una de estas tareas de forma consecuente con el modelo del usuario (el perfil de un diseñador «típico» de interiores) y con la percepción del sistema (lo que el diseñador de interiores espera de un sistema automatizado).

13.4 ACTIVIDADES DEL DISEÑO DE LA INTERFAZ

Una vez finalizado el análisis de tareas, quedan definidas detalladamente todas (u objetos y acciones) las que requiere el usuario final y comienza la actividad del diseño de la interfaz. Mediante el enfoque que se muestra a continuación se podrán llevar a cabo los primeros pasos del diseño de la interfaz [NOR86]:

1. Establecer los objetivos⁴ e intenciones para cada tarea.

¿Cuáles son los pasos que hay que seguir para llevar a cabo el diseño de la interfaz?

2. Hacer corresponder cada objetivo/intención con una secuencia de acciones específicas
3. Especificar la secuencia de acciones de tareas y subtareas, también llamado *escenario del usuario*, de la manera en que se ejecutarán a nivel de la interfaz.
4. Indicar el estado del sistema, esto es, el aspecto que tiene la interfaz cuando se está llevando a cabo el escenario del usuario.
5. Definir los mecanismo de control, esto es, los objetos y acciones disponibles para que el usuario altere el estado del sistema.

⁴ En muchos casos las actividades descritas en esta sección son llevadas a cabo por un ingeniero del software. Esperemos que el individuo tenga experiencia en ingeniería humana y en el diseño de la interfaz de usuario.

⁵ Entre los objetivos se pueden incluir la consideración de la utilidad de las tareas, la efectividad al llevar a cabo el objetivo comercial primordial, el grado de rapidez de aprendizaje de las tareas y el grado de satisfacción de los usuarios con la implementación final de la tarea.

6. Mostrar la forma en que los mecanismos de control afectan al estado del sistema.
7. Indicar la forma en que el usuario interpreta el estado del sistema a partir de la información proporcionada gracias a la interfaz.

Aunque el diseñador de la interfaz se guía por las reglas de oro abordadas en la Sección 15.1, deberá considerar la forma en que se va a implementar la interfaz, el entorno (por ejemplo, tecnología de pantalla, sistema operativo, herramientas de desarrollo) que se va a utilizar y otros elementos de la aplicación que «se encuentren por detrás» de la interfaz.

15.4.1. Definición de objetos y acciones de la interfaz

Un paso importante en el diseño de la interfaz es la definición de los objetos y acciones que se van a aplicar. Para llevar a cabo esta definición, el escenario del usuario se analiza sintácticamente de manera muy similar a como se analizaban las narrativas de procesamiento del Capítulo 12. Esto es, se escribe la descripción del escenario de un usuario. Los sustantivos (objetos) y los verbos (acciones) se aíslan para crear una lista de objetos y de acciones.

Referencia cruzada

En la Sección 12.6.2 se puede encontrar un estudio completa del análisis semántico gramatical.

Una vez que se han definido y elaborado iterativamente tanto los objetos como las acciones, se establecen categorías por tipos. Los objetos se identifican como objetos origen, destino y de aplicación. Un *objeto origen* (por ejemplo, un icono de informes) se arrastra y se coloca sobre otro *objeto destino* (por ejemplo, un icono de impresora). La implicación de esta acción es crear una copia impresa de un informe. Un *objeto de aplicación* representa los datos específicos de la aplicación que no se manipulan directamente como parte de la interacción de la pantalla. Por ejemplo, una lista de correo postal se utiliza para almacenar los nombres que se utilizarán para un correo postal. Esta lista se puede ordenar, fusionar o purgar (acciones basadas en menú) pero no se puede arrastrar y colocar mediante la interacción del usuario.



¿Qué es el formato de pantalla y cómo se aplica?

Una vez que el diseñador queda satisfecho con la definición de todos los objetos y acciones importantes (para una iteración de diseño), se lleva a cabo el *formato de*

pantalla. Al igual que otras actividades de diseño de la interfaz, el formato de pantalla es un proceso interactivo en donde se lleva a cabo el diseño gráfico y la colocación de los iconos, la definición del texto descriptivo en pantalla, la especificación y títulos para las ventanas, y la definición de los elementos del menú principales y secundarios. Si una metáfora con el mundo real es adecuada para la aplicación, queda especificada en ese momento y el formato se organiza para complementar esa metáfora.

Para mostrar la breve ilustración de los pasos de diseño descritos anteriormente, tomemos en consideración un escenario de usuario para la versión avanzada del sistema *HogarSeguro* (descrito en capítulos anteriores). En la versión avanzada, se puede acceder a *HogarSeguro* mediante módem o Internet. Una aplicación para PC permite que un propietario compruebe el estado de la casa desde una localización remota, reinicializar la configuración *HogarSeguro*, activar y desactivar el sistema, (empleando una opción de vídeo con un coste extra⁶), y supervisar visualmente las habitaciones dentro de la casa. A continuación, se muestra un escenario preliminar de usuario para la interfaz:

Referencia cruzada

El escenario descrito aquí es similar a los casos de estudio descritos en el Capítulo 11.

Escenario. El propietario de la casa desea acceder al sistema *HogarSeguro* instalado en su casa. Mediante el sistema operativo de un PC remoto (por ejemplo, un portátil que el propietario se lleve al trabajo o de viaje), el propietario determina el estado del sistema de alarma, arma o desarma el sistema, reconfigura las zonas de seguridad y observa las diferentes habitaciones de la casa mediante la preinstalación de una cámara de vídeo.

Para acceder a *HogarSeguro* desde una localización remota, el propietario proporciona un identificador y una contraseña. Con esto se definen los niveles de acceso (por ejemplo, todos los usuarios puede que **no** tengan la capacidad de reconfigurar el sistema) y se proporciona seguridad. Una vez validados, el usuario (con **los** privilegios para el acceso) comprueba el estado del sistema, y cambia el estado armando y desarmando *HogarSeguro*. El usuario reconfigura el sistema visualizando todas las zonas configuradas actualmente, modificando las zonas cuando se requiera. El usuario observa el interior de la casa mediante cámaras de vídeo estratégicamente ubicadas. El usuario puede utilizar las cámaras para recorrer todo el interior y ampliarlo para ofrecer diferentes visiones del interior de la casa.

Tareas del propietario:

- *acceder* al sistema *HogarSeguro*;
- *introducir* un ID y una contraseña para permitir un acceso remoto;
- *comprobar* el estado del sistema;
- *activar* o *desactivar* el sistema *HogarSeguro*;

⁶ La opción de vídeo posibilita al usuario colocar una cámara de vídeo en lugares clave por la casa y examinar la salida desde una localización remota. ¿El Gran Hermano?

- *visualizar* el plano de la casa y las localizaciones de los **sensores**;
- *visualizar* zonas en el plano de la casa;
- *cambiar* zonas en el plano de la casa;
- *visualizar* las localizaciones de las cámaras de vídeo en el plano de la casa;
- *seleccionar* la cámara de vídeo para tener visión;
- *observar* las imágenes de vídeo (cuatro encuadres por segundo);
- *recorrer* y *ampliar* las habitaciones con la cámara de vídeo.

Los objetos (negrita) y las acciones (cursiva) se extraen de la lista de tareas del propietario descritas anteriormente. La mayoría de los objetos anteriores son objetos de aplicaciones. Sin embargo la localización de la cámara de vídeo (un objeto origen) se arrastra y se coloca sobre la cámara de vídeo (un objeto destino) para crear una imagen de vídeo (una ventana con la presentación de un vídeo).

Para la supervisión del vídeo se crea un esbozo del formato de pantalla (Fig. 15.2). Para invocar la imagen de vídeo, se selecciona un icono de **localización de cámara de vídeo C**, ubicado en el **plano de la casa** que se visualiza en la **ventana de supervisión**. En este caso, la localización de una cámara en la sala de estar (SE), se arrastra y se coloca sobre el icono de vídeo de cámara en la parte superior izquierda de la pantalla. Entonces, mediante la visualización del recorrido realizado por el vídeo desde la cámara localizada en la sala de estar (SE), aparece la **ventana de imagen de vídeo**. Las diapositivas de control del **recorrido** por las habitaciones y de las **ampliaciones** se utilizan para controlar la amplitud y la dirección de la imagen de vídeo. Para

seleccionar una visión desde otra cámara, el usuario simplemente arrastra y coloca el icono de **localización de cámara** dentro del icono **cámara** del ángulo superior izquierdo de la pantalla.

Esta muestra de esbozo de formato tendría que ir complementado mediante una ampliación de todos los elementos del menú dentro de la barra de menú, indicando las acciones disponibles para el (estado) *modo de supervisión del vídeo*. Durante el diseño de la interfaz se debería crear un conjunto completo de esbozos para todas y cada una de las tareas del propietario descritas en el escenario del usuario.

15.4.2. Problemas del diseño

A medida que la interfaz de usuario va evolucionando casi siempre afloran cuatro temas comunes de diseño: el tiempo de respuesta del sistema, los servicios de ayuda al usuario, la manipulación de información de errores y el etiquetado de órdenes. Desgraciadamente, muchos diseñadores no abordan estos temas dentro del proceso de diseño hasta que es relativamente tarde (algunas veces no se siente la aparición de un error hasta que se dispone del prototipo operativo). El resultado suele ser una iteración innecesaria, demoras de proyecto y frustración del usuario. Es infinitamente mejor establecer el tema de diseño que se vaya a tener en cuenta al iniciar el diseño del software, es decir cuando los cambios son fáciles y los costes más reducidos.

Para muchas aplicaciones interactivas el tiempo de respuesta del sistema es el principal motivo de queja. En general, el tiempo de respuesta del sistema se mide desde el punto de vista que el usuario realiza la acción

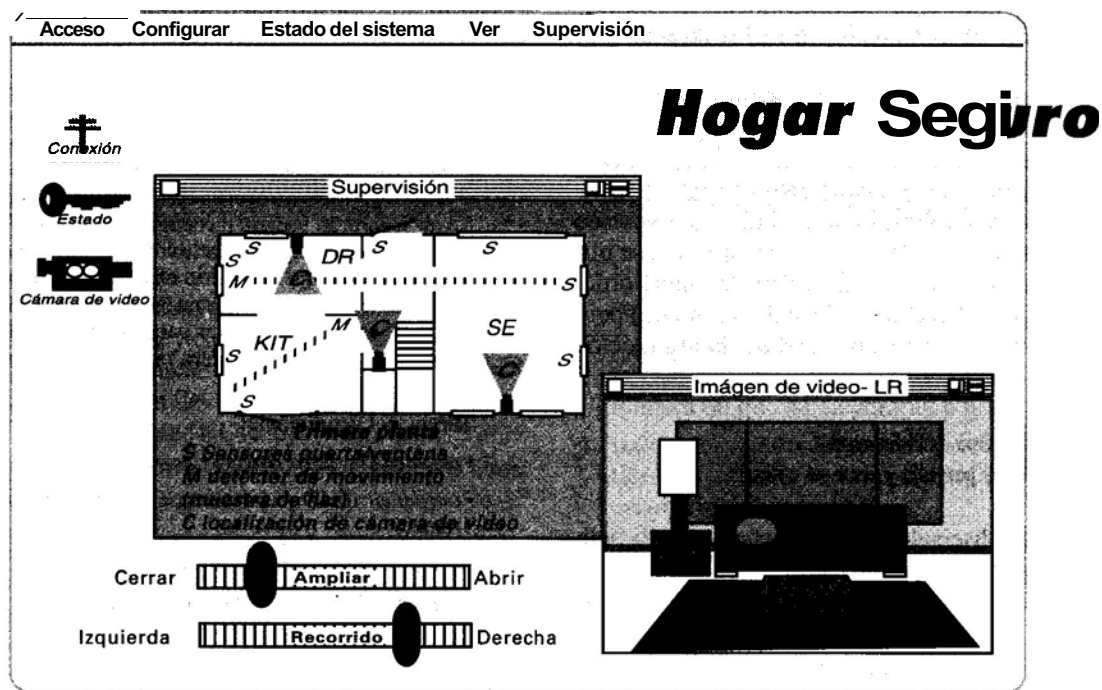


FIGURA 15.2. Formato preliminar de pantalla.

de control (por ejemplo, pulsar la tecla intro o pulsar el botón del ratón) hasta que el software responde con la salida o acción deseada.

El tiempo de respuesta del sistema tiene dos características importantes: la duración y la variabilidad. Si la *duración* de la respuesta del sistema es demasiado larga, es inevitable obtener como resultado la frustración y el estrés del usuario. Sin embargo, si la interfaz va marcando el ritmo del usuario una duración breve del tiempo de respuesta puede ser también perjudicial. Un tiempo de respuesta rápido puede obligar a que el usuario se precipite y cometa errores.



Si no se puede evitar una respuesta variable, asegúrese de proporcionar alguna indicación visual del progreso para que el usuario sepa lo que está ocurriendo.

La *variabilidad* se refiere a la desviación del tiempo de respuesta promedio, y en muchos aspectos es la característica **más** importante del tiempo de respuesta. Una variabilidad baja posibilita al usuario establecer un ritmo de interacción, aunque el tiempo de respuesta sea relativamente largo. Por ejemplo, es preferible obtener una segunda respuesta de una orden a una respuesta que varíe de 0,1 a 2,5 segundos. El usuario siempre estará desconcertado y preguntándose si ha ocurrido algo «diferente» detrás de la escena.

Casi todos los usuarios de un sistema interactivo basado en computadora requieren ayuda, ahora y siempre. Los dos tipos de funciones de ayuda más comunes son: **integradas** y **complementarias** (añadibles). [RUB88]. Se *diseña una función de ayuda integrada* dentro del mismo software desde el principio. Suele ser sensible al contexto, lo que posibilita al usuario seleccionar entre los temas que sean relevantes para las acciones que esté llevando a cabo en ese momento. Obviamente esto reduce el tiempo que requiere para obtener ayuda, e incrementa su «familiaridad» con la interfaz. *Una función de ayuda complementaria* se añade al software una vez construido el sistema. En muchos aspectos es muy similar a un manual de usuario en línea con una capacidad limitada de consulta. Es posible que el usuario tenga que buscar en una lista de miles de temas para encontrar la guía adecuada, entrando normalmente en las ayudas incorrectas y recibiendo mucha información irrelevante. No hay ninguna duda de que es preferible el enfoque de funciones de ayuda integradas al enfoque de funciones complementarias.



¿Cuáles son los temas de diseño que deberán tenerse en cuenta en la construcción de las funciones de ayuda?

Cuando se va a considerar un servicio de ayuda hay una serie de temas de diseño que deberán abordarse [RUBSS]:

- **¿Se** necesitará disponer de todas las funciones del sistema en cualquier momento durante la interacción del sistema? Opciones: ayuda solo para un subconjunto de todas las funciones y acciones; ayuda para todas las funciones.
- **¿De** qué forma solicitará ayuda el usuario? Opciones: un menú de ayuda; una tecla de función especial; una orden de AYUDA.
- **¿Cómo** se representará la ayuda? Opciones: una ventana separada; una referencia a un documento impreso (no es lo ideal); una sugerencia de una o dos líneas que surge en una localización fija en la pantalla.
- **¿Cómo** regresará el usuario a la interacción normal? Opciones: un botón de retorno visualizado en la pantalla; una tecla de función o una secuencia de control.
- **¿Cómo** se estructurará la información sobre la pantalla? Opciones: una estructura «plana» donde el acceso a la información se realiza mediante una contraseña; una jerarquía estratificada de información que va proporcionando más datos a medida que el usuario va entrando por la estructura; la utilización de hipertexto.

Cuando ha salido algo mal, los mensajes de error y las sugerencias son «malas noticias» para los usuarios de sistemas interactivos. En el peor de los casos, estos mensajes imparten información sin utilizar o engañosa y sirven solo para incrementar la frustración del usuario. Existen muy pocos usuarios que puedan decir que no se han encontrado con un error del tipo:

FALLO GRAVE DEL SISTEMA - - 14A

En algún lugar debe existir una explicación del error 14A, o sino ¿por qué habrá incluido el diseñador esta identificación? A pesar de esto, el mensaje de error no proporciona una indicación verdadera de lo que va mal o de donde mirar para obtener más información. Un mensaje de error como el que se ha presentado anteriormente no hace nada por aliviar la ansiedad del usuario o por ayudar a solucionar el problema.

En general, todos los mensajes de error o sugerencias de un sistema interactivo deberán tener las características siguientes:

- El mensaje deberá describir el problema en una jerga que el usuario pueda entender.
- El mensaje deberá proporcionar consejos constructivos para recuperarse de un error.
- El mensaje deberá indicar cualquier consecuencia negativa del error (por ejemplo, los archivos de datos posiblemente deteriorados) para que el usuario pueda comprobar y garantizar que no hay ninguno (y corregirlos si existen).



Duplica el esfuerzo y las palabras al solucionar errores cuando piense que necesita una función de ayuda y, de esta forma, es probable que consiga un buen resultado.

- El mensaje deberá ir seguido por una clave audible o visual. Esto es, para acompañar la visualización del mensaje se podría generar un pitido, o aparecer momentáneamente una luz destelleante o visualizarse en un color que se pueda reconocer fácilmente como el «color del error».
- El mensaje no deberá tener «sentencias». Esto es, las palabras del mensaje nunca deberán culpar al usuario.

Dado que a nadie le gusta realmente tener malas noticias, a muy pocos usuarios les gustará tener un mensaje de error independientemente del diseño. No obstante, una filosofía eficaz de mensajes de error puede ayudar mucho a la hora de mejorar la calidad de un sistema interactivo y reducirá significativamente la frustración del usuario cuando aparecen problemas.

Anteriormente las órdenes escritas o tecleadas eran el modo de interacción más común entre el usuario y el

software del sistema, y se utilizaban normalmente para aplicaciones de todo tipo. Actualmente, la utilización de interfaces orientadas a ventanas en donde solo se señala y se selecciona, ha reducido el hecho de depender de órdenes tecleadas, aunque muchos usuarios avanzados siguen prefiriendo el modo de interacción orientado a órdenes. Cuando se proporcionan Órdenes escritas como modo de interacción surge una serie de temas de diseño:

- ¿Se corresponden todas las opciones del menú con su órdenes?
- ¿Qué formas adquirirán las órdenes? Opciones: una secuencia de control (por ejemplo, alt-P); teclas de función; una palabra tecleada.
- ¿Será difícil aprender y recordar las órdenes? ¿Qué se puede hacer si se olvida una orden?
- ¿Podrá personalizar o abreviar el usuario las órdenes?

15.5 HERRAMIENTAS DE IMPLEMENTACIÓN

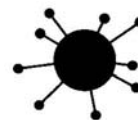
Una vez creado el modelo de diseño, se implementa como un prototipo⁷, que los usuarios han examinado (aquellos que adaptan el modelo del usuario descrito anteriormente), y que se ha basado en los comentarios de los usuarios.

Para acoplar este enfoque de diseño iterativo se ha desarrollado una clase extensa de herramientas diseño de interfaz y de generación de prototipos. Estas herramientas así llamadas, *juego de herramientas de la interfaz de usuario o sistemas de desarrollo de la interfaz de usuario (SDIU)*, proporcionan componentes u objetos que facilitan la creación de ventanas, menús, interacción de dispositivos, mensajes de error, Órdenes y muchos otros elementos de un entorno interactivo.

Mediante los componentes de software preestablecidos que se pueden utilizar para crear una interfaz de usuario, un SDIU proporcionará los mecanismos [MYE89] para:

- gestionar los dispositivos de salida (tales como el ratón o el teclado);
- validar la entrada del usuario;
- manipular los errores y visualizar mensajes de error;

- proporcionar una respuesta (por ejemplo, un eco automático de la entrada)
- proporcionar ayuda e indicaciones de solicitud de entrada de órdenes;
- manipular ventanas y campos, desplazarse por las ventanas;
- establecer conexiones entre el software de la aplicación y la interfaz;
- aislar la aplicación de las funciones de gestión de la interfaz;
- permitir que el usuario personalice la interfaz.



Diseño de la Interfaz de usuario

Estas funciones se pueden implementar mediante un enfoque gráfico o basado en lenguajes.

15.6 EVALUACIÓN DEL DISEÑO

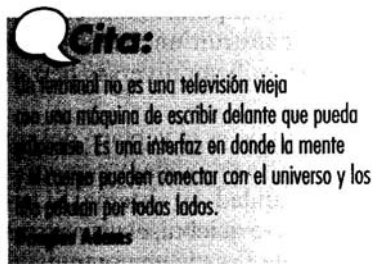
Una vez que se ha creado un prototipo de interfaz de usuario, deberá sufrir una evaluación para determinar si cumple las necesidades del usuario. La evaluación podrá abarcar un espectro de formalidad: desde «pruebas» informales en donde el usuario proporciona respuestas espon-

táneas hasta un estudio formalmente diseñado que utilizará métodos estadísticos para la evaluación de cuestionarios cumplimentados por un grupo de usuarios finales.

El ciclo de evaluación de la interfaz adquiere forma en la Figura 15.3. Una vez finalizado el modelo de dise-

⁷ Debe destacarse que en algunos casos (por ejemplo, los indicadores del panel de los aviones), el primer paso debena ser simular la interfaz en un dispositivo en vez de utilizar el hardware del indicador.

ño, se crea un prototipo de primer nivel. Este prototipo es evaluado por el usuario, que es quien proporcionará al diseñador los comentarios directos sobre la eficacia de la interfaz. Además, si se utilizan técnicas formales de evaluación (por ejemplo, cuestionarios, hojas de evaluación), es posible que el diseñador extraiga información de estos datos (por ejemplo, el 80 por 100 de los usuarios no mostró afinidad con el mecanismo para grabar archivos de datos). Las modificaciones que se realicen sobre el diseño se basarán en la entrada del usuario y entonces se creará el prototipo de segundo nivel. El ciclo de evaluación continúa hasta que ya no sean necesarias más modificaciones del diseño de la interfaz.



El enfoque de generación de prototipos es eficaz, ahora bien ¿es posible evaluar la calidad de la interfaz de usuario antes de construir un prototipo? Si los problemas se pueden descubrir y solucionar rápidamente, el número de bucles en el ciclo de evaluación se reducirá y el tiempo de desarrollo se acortará. Si se ha creado un modelo de diseño de la interfaz, durante las primeras revisiones del diseño se podrán aplicar una serie de criterios [MOR81] de evaluación:

1. La duración y la complejidad de la especificación que se haya escrito del sistema y de su interfaz proporcionan una indicación de la cantidad de aprendizaje que requieren los usuarios del sistema.
2. La cantidad de **tareas** especificadas y la cantidad media de acciones por tarea proporcionan una indicación del tiempo y de la eficacia global del sistema.
3. La cantidad de acciones, tareas y estados de sistemas indicados con el modelo de diseño indican la carga de memoria que tienen los usuarios del sistema.
4. El estilo de la interfaz, las funciones de ayuda y el protocolo de solución de errores proporcionan una indicación general de la complejidad de la interfaz y el grado de aceptación por parte del usuario.

Una vez construido el primer prototipo, el diseñador puede recopilar una diversidad de datos cualitativos y cualitativos que ayudarán a evaluar la interfaz. Para recopilar los datos cualitativos, se pueden distribuir cuestionarios a los usuarios del prototipo. Las preguntas pueden ser (1) del tipo de respuesta si/no; (2) respuesta numérica; (3) respuesta con escala de valoración (subjetiva), o (4) respuesta por porcentajes (subjetiva). A continuación se muestran unos ejemplos:

1. ¿Eran claros los iconos? En caso negativo, ¿Qué iconos no eran claros?
2. ¿Eran fáciles de recordar y de invocar las acciones?
3. ¿Cuántas acciones diferentes ha utilizado?
4. ¿Resultaron fáciles de aprender las operaciones básicas del sistema? (valoración de 1 a 5)
5. En comparación con otras interfaces que haya utilizado, ¿Cómo evaluaría ésta?

entre el 1 % mejores, 10% mejores, 25% mejores, 50% mejores, 50% inferiores.



Interfaz de usuario.

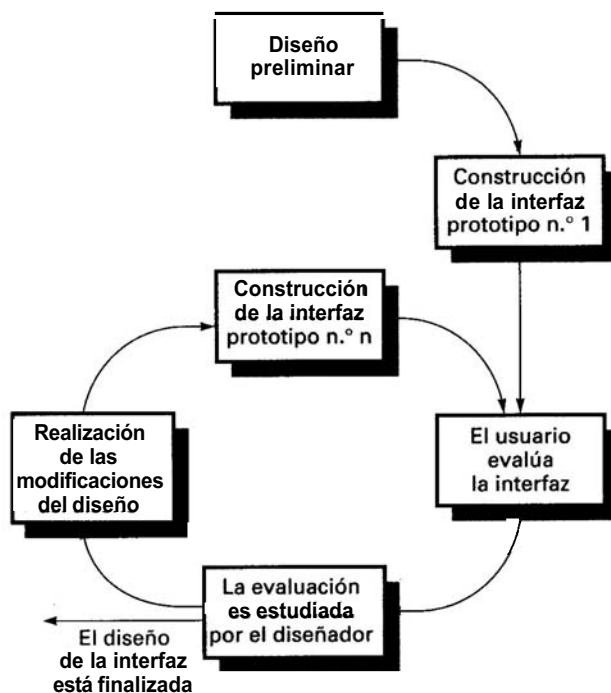


FIGURA 15.3. El ciclo de evaluación de diseño de la interfaz.

Si se desea obtener datos cuantitativos, se puede llevar a cabo una forma de análisis para el estudio del tiempo. Los usuarios son observados durante la interacción; y para tener una guía durante la modificación de la interacción se recopilan y utilizan datos tales como: grupo de tareas finalizadas correctamente por encima del período de tiempo estándar; frecuencia de acciones; secuencia de acciones; tiempo transcurrido «mirando» la pantalla; número y tipo de errores, y tiempo de solución de errores; tiempo necesario para utilizar la ayuda; y cantidad de referencias de ayuda por período de tiempo estándar.

Un estudio completo de los métodos de evaluación de la interfaz de usuario queda fuera del ámbito de este libro. Para más información, véase [LEA88] y [MAN97].

RESUMEN

Se puede argumentar que la interfaz de usuario es el elemento más importante de un sistema o producto basado en computadora. Si la interfaz tiene un diseño pobre, la capacidad que tiene el usuario de aprovecharse de la potencia de proceso de una aplicación se puede dificultar gravemente. En efecto, una interfaz débil puede llevar al fracaso de una aplicación con una implementación sólida y un buen diseño.

Existen tres principios importantes que dirigen el diseño de interfaces de usuario eficaces: (1) poner el control en manos del usuario; (2) reducir la carga de la memoria del usuario; (3) construir una interfaz consecuente. Para lograr que una interfaz se atenga a estos principios, se deberá desarrollar un proceso de diseño organizado.

El diseño de la interfaz de usuario comienza con la identificación de los requisitos del usuario, de las tareas y del entorno. El análisis de tareas es una actividad de diseño que define las tareas y acciones del usuario empleando un enfoque elaborativo u orientado a objetos.

Una vez que se han definido las tareas, los escenarios del usuario se crean y analizan para definir un conjunto de objetos y acciones de la interfaz. Esto es lo que proporciona la base para la creación del formato de la pantalla, el cual representa el diseño gráfico y la colocación de iconos, la definición de un texto descriptivo en pantalla, la especificación y titulación de ventanas y la especificación de los elementos importantes y secundarios del menú. Cuando se va a refinar un modelo de diseño para el sistema se tienen en consideración temas de diseño tales como tiempo de respuesta, estructura de Órdenes y acciones, manipulación de errores y funciones de ayuda. Para construir un prototipo que el usuario pueda evaluar se utilizan diversas herramientas de implementación.

La interfaz de usuario es la ventana del software. En muchos casos, la interfaz modela la percepción que tiene un usuario de la calidad del sistema. Si la ventana se difumina, se ondula o se quiebra, el usuario puede rechazar un sistema potente basado en computadora.

REFERENCIAS

- [DUM88] Dumas, J.S., *Designing User Interfaces for Software*, Prentice-Hall, 1988.
- [LEA88] Lea, M., «Evaluating User Interfaces Designs», *User Interface Design for Computer Systems*, Halstead Press (Wiley), 1988.
- [MAN97] Mandel, T., *The Elements of User Interface Design*, Wiley, 1997.
- [MON84] Monk, A. (ed), *Fundamentals of Human-Computer Interaction*, Academic Press, 1984.
- [MOR81] Moran, T.P., «The Command Language Grammar: A Representation for the User Interface of Interactive Computer Systems», *Intl. Journal of Man-Machine Studies*, vol. 15, pp. 3-50.
- [MYE89] Myers, B.A., «User Interface Tools: Introduction and Survey», *IEEE Software*, Enero 1989, pp. 15-23.
- [NOR86] Norman, D.A., «Cognitive Engineering», *User-Centered Systems Design*, Lawrence Earlbaum Associates, Nueva Jersey, 1984.
- [RUB88] Rubin, T., *User Interface Design for Computer Systems*, Haldstead Press (Wiley), 1988.
- [SHN87] Shneiderman, B., *Designing the User Interface*, Addison-Wesley, 1987.

PROBLEMAS Y PUNTOS A CONSIDERAR

15.1. Describa la peor interfaz con la que haya trabajado alguna vez y critíquela en relación con los conceptos que se han presentado en este capítulo. Describa la mejor interfaz con la que haya trabajado alguna vez y critíquela en relación con los conceptos presentados en este capítulo.

15.2. Desarrolle dos principios de diseño más que «den el control al usuario».

15.3. Desarrolle dos principios de diseño más que «reduzcan la carga de memoria del usuario».

15.4. Desarrolle dos principios de diseño más que «ayuden a construir una interfaz consecuente».

15.5. Considere una de las aplicaciones interactivas siguientes (o una aplicación asignada por su profesor):

- un sistema de autoedición
- un sistema de diseño asistido por computadora
- un sistema de diseño de interiores
- un sistema de matriculación automatizado para la universidad
- un sistema de gestión de biblioteca
- un sistema de votación basada en Internet para las elecciones públicas
- un sistema bancario en casa
- una aplicación interactiva asignada por su instructor

Desarrolle un modelo de diseño, un modelo de usuario, una imagen de sistema y una percepción de sistema para cualquiera de los sistemas anteriores.

15.6. Realice un análisis detallado de tareas para cualquiera de los sistemas que se relacionan en el Problema 15.5. Utilice un enfoque elaborativo u orientado a objetos.

15.7. Como continuación al Problema 15.6, defina objetos y acciones para la aplicación que acaba de seleccionar. Identifique todos los tipos de objetos.

15.8. Desarrolle un conjunto de formatos de pantalla con una definición de los elementos del menú principales y secundarios para el sistema que haya elegido en el Problema 15.5.

15.9. Desarrolle un conjunto de formatos de pantalla con una definición de los elementos principales y secundarios del menú para el sistema estándar *HogarSeguro* de la Sección 15.4.1. Puede optar por un enfoque diferente al mostrado en la Figura 15.2 sobre un formato de pantalla.

15.10. Describa el enfoque utilizado en las funciones de ayuda del usuario que haya utilizado para el modelo de diseño en

el análisis de tareas que haya llevado a cabo desde el Problema 15.6 al 15.8.

15.11. Proporcione unos cuantos ejemplos que muestren la razón de que la variabilidad del tiempo de respuesta pueda considerarse un tema a tener en cuenta.

15.12. Desarrolle un enfoque que integre automáticamente los mensajes de error y una función de ayuda al usuario. Esto es, que el sistema reconozca automáticamente el tipo de error y proporcione una ventana de ayuda con sugerencias para corregirlo. Realice un diseño de software razonablemente completo que tenga en consideración estructuras de datos y algoritmos.

15.13. Desarrolle un cuestionario de evaluación de la interfaz que contenga 20 preguntas genéricas que se puedan aplicar a la mayoría de las interfaces. Haga que 10 compañeros de clase rellenen el cuestionario del sistema de interfaz que vayan a utilizar todos. Resuma los resultados e informe de ellos a la clase.

RECOMENDACIONES Y FUENTES DE INFORMACIÓN

Aunque el libro de Donald Norman no trata específicamente interfaces hombre-máquina, mucho de lo que su libro (*The Design of Everyday Things*, Reissue edition, Currency/Doubleday, 1990) tiene que decir sobre la psicología de un diseño eficaz se aplicará a la interfaz de usuario. Es una lectura recomendada para todos los que sean serios a la hora de construir un diseño de interfaz de alta calidad.

Durante la década pasada se han escrito docenas de libros acerca del diseño de interfaces. Sin embargo, los libros de Mandel [MAN97] y Shneiderman (*Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 3.^a ed., Addison-Wesley, 1990) continúan proporcionando el estudio más extenso acerca de la materia. Donnelly (*In Your Face: The Best of Interactive Design*, Rockport Publications, 1998), Fowler, Stanwick y Smith (*GUI Design Handbook*, McGraw-Hill, 1998), Weinschenk, Jamar, y Yeo (*GUI Design Essentials*, Wiley, 1997), Galitz (*The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*, Wiley, 1996), Mullet y Sano (*Designing Visual Interfaces: Communication Oriented Techniques*, Prentice Hall, 1995), y Cooper (*About Face: The Essentials of User Interface Design*, IDG Books, 1995) han escrito estudios que proporcionan las líneas generales y principios de diseño adicionales así como las sugerencias para la elicitación de requisitos, modelado, implementación y comprobación del diseño.

El análisis y modelado de tareas son las actividades fundamentales del diseño de la interfaz. Hackos y Redish (*User and Task Analysis for Interface Design*, Wiley, 1998) han escrito un libro especializado en estos temas y proporcionan un método detallado para abordar el análisis de tareas. Wood (*User Interface Design: Bridging the Gap from User Requirements to Design*, CRC Press, 1997) tiene en consideración la actividad de análisis para interfaces y la transición hacia las tareas de diseño. Uno de los primeros libros que presentan el tema de los escenarios en el diseño de la interfaz de

usuario ha sido editado por Carroll (*Scenario-Based Design: Envisioning Work and Technology in System Development*, Wiley, 1995). Horrocks (*Constructing the User Interface with Statecharts*, Addison-Wesley, 1998) ha desarrollado un método formal para el diseño de interfaces de usuario que se basa en el modelado del comportamiento basado en el estado.

La actividad de evaluación se centra en la usabilidad. Los libros de Rubin (*Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*, Wiley, 1994) y Nielson (*Usability Inspection Methods*, Wiley, 1994) aborda el tema de forma considerable y detallada.

La computadora Apple de Macintosh popularizó las interfaces de usuario con diseños sólidos y fáciles de utilizar. El personal de Apple (*Macintosh Human Interface Guidelines*, Addison-Wesley, 1993) estudia la apariencia y utilización del ahora famoso Macintosh (y tan imitado). Uno de los primeros libros que se han escrito acerca de la interfaz de Microsoft Windows fue elaborado por el personal de Microsoft (*The Windows Guidelines for Software Design: An Application Design Guide*, Microsoft Press, 1995).

En un libro de Murphy (*Front Panel: Designing Software for Embedded User Interfaces*, R&D Books, 1998), el cual puede resultar de gran interés para los diseñadores del producto, se proporciona una guía detallada para el diseño de interfaces de sistemas empotrados y aborda los peligros inherentes en controles, en manipulación de maquinaria pesada e interfaces para sistemas médicos y de transporte. El diseño de la interfaz para productos empotrados también se estudia en el libro de Garrett (*Advanced Instrumentation and Computer I/O Design: Real-Time System Computer Interface Engineering*, IEEE, 1994).

Una amplia variedad de fuentes de información sobre el diseño de la interfaz de usuario y de temas relacionados están disponibles en Internet. Una lista actualizada de referencias relevantes para la interfaz de usuario se puede encontrar en <http://www.pressman5.com>