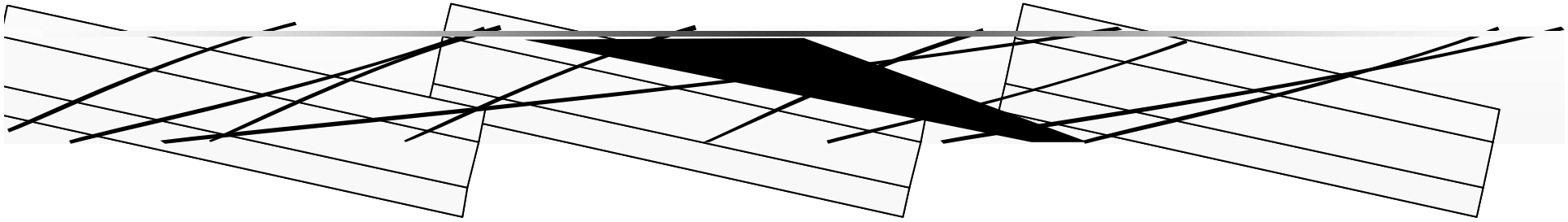
Diseño de Interfaces Hombre-Máquina (HMI)



Alfredo Rosado

SID. ITT-SE

Objetivos de la sesión

- Tipos de Interfaces gráficas
- Teorías y modelos de alto nivel
- Principios y reglas: diálogos y prevención de errores.
- Principios generales:
  - Guías, estándares, métodos de prototipos y pruebas

Interfaces gráficas

Los tres estilos más comunes de interfaces gráficas hombre computadora son:

- WYSIWYG What you see is what you get, Lo que tú ves es lo que puedes conseguir

- Manipulación directa e Interfaces de usuario basados en iconos.

- Se mencionarán brevemente algunos otros estilos de interacción como: menú de selección, lenguaje de comandos, lenguaje natural, diálogo de pregunta-respuesta.

Estos estilos de interacción no son esencialmente gráficos pero pueden ser utilizados en aplicaciones gráficas.

Lo que tú ves es lo que puedes conseguir (WYSIWYG)

- En una interfaz WYSIWYG la representación gráfica con la cual los usuarios interactúan en la pantalla es esencialmente la misma imagen creada por la aplicación. Muchas de la aplicaciones gráficas tienen algún componente WYSIWYG.
- Existen muchos editores que tienen interfaces WYSIWYG; principalmente aquellas aplicaciones gráficas. Lo que nosotros vemos en la pantalla es el resultado de nuestra aplicación en papel.

Manipulación directa

- La interfaz gráfica de manipulación directa es aquella en la cual a través de acciones físicas los usuarios manipulan los objetos gráficos (son hechos de acuerdo con entidades y experiencias del mundo real. Por ejemplo: metáforas del mundo real) en la pantalla. Estas acciones son ejecutadas comúnmente utilizando el ratón.
- Shneiderman [1992] propuso en su terminología los siguientes principios para describir la manipulación directa:
- Continua representación de los objetos y acciones de interés.
- Acciones físicas o presión de botones etiquetados en lugar de sintaxis muy compleja.
- Rápidas y reversibles operaciones que afectan a los objetos de interés son inmediatamente visibles.
- Con el uso de manipulación directa se puede evitar el uso de sintaxis compleja, por lo tanto, se reducen los errores cometidos con el uso del mismo, ayudando a los novatos a aprender más rápido la funcionalidad básica del sistema. Por ejemplo, para borrar un archivo dentro de la interfaz de Macintosh se selecciona visualmente el archivo y se arrastra hasta el bote de basura.

Manipulación directa

- Comúnmente manipulación directa es utilizado con otros estilos de interfaces como la ejecución de comandos por medio de menús o por el teclado permitiendo una mejor usabilidad de la misma.
- Manipulación directa es un estilo de interfaz muy fácil de aprender y muy poderosa, muchas veces es presentada como el mejor estilo de interfaz, pero sin embargo tiene algunas limitaciones.
- La exactitud dentro de la manipulación es dejada mucha veces a la habilidad que tenga el usuario con el uso del ratón (eje. ajustar botón movible en un valor especificado).
- Requiere de mucho conocimiento del mundo real (eje. es necesario que una persona sepa utilizar una calculadora común y corriente para que se le facilite el uso de la que está representada en la pantalla).
- La velocidad en usuario experimentados es reducida si el usuario es forzado a utilizar solo manipulación directa cuando otro tipo de interacción es más rápida (p. eje. para borrar todos los archivos con extensión “txt” -en una carpeta- probablemente se requiera encontrarlos visualmente dentro de muchos iconos y arrastrarlos hasta el bote de basura siendo más fácil borrarlos como en DOS con un comando “del *.txt”).

Interfaces de usuario basados en iconos

- *Este estilo de interfaz se basa en gráficas (iconos) para representar un objeto, una acción, una propiedad o algún otro concepto. No siempre el uso de iconos es mejor para representar algún concepto o acción, muchas veces es mejor utilizar texto para representar algo.*
- *No se puede decir que el icono sea mejor que el texto o viceversa, pero sí se puede decir que los iconos tienen las siguientes ventajas:*
- *iconos bien diseñados pueden reconocerse más rápido que las palabras, toman menos espacio en la pantalla, si se toma mucho cuidado en el diseño de iconos, puede ser un lenguaje independiente, permitiendo a la interfaz ser usada en diferentes países.*

Otros estilos de interacción

Los tres estilos mostrados anteriormente pueden ser llamados “basado en gráficos”.

Existen otras formas de interacción que no son “basados en gráficos”:

- menú de selección,
- lenguaje de comandos,
- lenguaje natural,
- diálogo de pregunta respuesta

menú de selección

- Los menús pueden ser usados en aplicaciones gráficas y no gráficas.
- La principal ventaja que tienen es que cuando se está trabajando, con solo ver texto o un icono en un menú se puede reconocer la palabra y significado y así recordar su función (al ver la palabra “abrir” en el menú archivo, se puede abrir un documento) a diferencia de tener que recordar toda la instrucción con su sintaxis. Este tipo de interacción es muy atractivo para nuevos usuarios ya que no tienen que memorizar muchos comandos ni sintaxis.

El lenguaje de comandos

El lenguaje de comandos es la tradicional manera de interactuar con la computadora. Con esta técnica es fácil de extender (solamente hay que agregar un nuevo comando). Es bastante rápida de usar para usuarios con experiencia. Los errores son más probables con lenguaje de comandos que con menús.

Por ejemplo:

La siguiente sentencia (comando del sistema operativo DOS) borra el archivo datos.dat del disco.

C:\>delete datos.dat

El lenguaje natural

El lenguaje natural frecuentemente es propuesto como el último objetivo para los sistemas interactivos.

Si la computadora es capaz de entender nuestros comandos en inglés (hablados o escritos) cualquier persona puede ser capaz de usarlos. Sin embargo, el reconocer la voz implica muchos problemas así como el escribir largas sentencias es muy tedioso. También, como el lenguaje natural no limita el conjunto de comandos que debe manipular este puede ser bastante ambiguo

Diálogo de pregunta-respuesta

En el diálogo de pregunta-respuesta la computadora inicia haciendo una pregunta y el usuario responde a esta por medio del teclado. Si el conjunto de respuestas es pequeño se puede dar la alternativa de seleccionar de un menú la opción en lugar de teclearla. Un problema de este tipo de interacción es la incapacidad para ir algunos pasos atrás para corregir una respuesta, mayor aún, solamente se tienen el contexto de la pregunta para contestarla.

Por ejemplo:

¿Cual es la dirección del empleado?

Aquí no se sabe si hay que incluir, la colonia o no.

Teorías y modelos de alto nivel

- Vienen de la psicología y pedagogía
  - Buscan describir los factores que influyen el buen o mal resultado de un diseño
- ¿Cómo reacciona un ser humano ante una máquina?

TEORÍA DE LOS CUATROS NIVELES

La persona razona en cuatro niveles, es decir, tienen cuatro niveles de abstracción:

Estos niveles de abstracción son:

1. Léxico.- Es el lenguaje que hablamos. Ejemplo: A, B,C, etc. Aprendemos que el símbolo A es una "A"
2. Sintáctico.- Combinar para hacer palabras. Ejemplo: casa, Avión, etc. Combinamos letras para hacer palabras.
3. Semántico.- Es el significado de las palabras. Ejemplo: casa.- Lugar donde viven las personas.
4. Conceptual.- Es la idea, lo que tengo en mente. Ejemplo: Esa casa es muy bonita.

TEORÍA DE LOS CUATROS NIVELES

En nivel de interfaces lo podemos ver de la siguiente forma:

4. Conceptual.- Es la idea del sistema como un todo (Un editor de texto, un editor de gráficos, etc.)
3. Semántico.- Es el significado de:
   Pantallas
   Instrucciones
2. Sintáctico.- Es la unidad de:
   Comandos
   Secuencias
1. Léxico.- Son las dependencias con la computadora

Este modelo está estrechamente relacionado con la programación TOP-DOWN.

Modelo GOMS

- Este modelo, propuesto por Card, Moran y Newell, se basa en la clasificación de 4 niveles, de donde toma su nombre:

- **Metas** (**G**oals) del usuario

- **Operadores** (**O**perators) con los que cuenta la interfaz para dar servicio al usuario

- **Métodos** (**M**ethods), que indican cómo combinar los operadores para conseguir las metas

- **Reglas de selección** (**S**election Rules), que indican cómo hace el usuario para aplicar los operadores.

Interfaces gráficas con el usuario

- Precursores
  - PARC XEROX
  - Lisa/Macintosh
  - GEOS

Interfaces gráficas con el usuario

- GUI extendidos en el mercado.
  - Microsoft Windows. (3.11, 95 y NT)
  - Macintosh.
  - X Windows - Motif.
  - OS/2.
  - Open Look.

Principales elementos de control comunes

- Ventanas.
- Iconos.
- Menúes.
- Diálogos.
- Botones de acción.
  - Con icono o sin él.

Principales elementos de control comunes

- Botones de radio.
- Cuadros con marcas (Check boxes)
- Campos de edición.
- Apuntador del ratón.
- Barras de desplazamiento.
- Listas de selección.

Principales elementos de control no comunes

- Barras de herramientas.
- Diálogos en Carpetas (Tabbed dialogs)
- Menú contextual.
- Ayuda en línea.
- Campos de selección/edición (Combo boxes)

Principales aplicaciones comunes

- Panel de control.
- Administrador de archivos/programas = "Shell"
- Administrador de impresión = "Spooler"
- Calculadora.

Guías y estándares

- Existen de acuerdo al GUI subyacente
- Ejemplos:
  - Libros de Guías Mac
  - Guías Windows

Diseño de diálogos

8 “guías de oro”

- Consistencia
- Atajos para usuarios frecuentes
- Retroalimentación
- Diálogos Cerrados

INTERFAZ CON EL USUARIO

La tarea de un equipo de desarrollo de software es imaginárselas para ocultarle al usuario del sistema las complejidades del mismo.

OBJETIVOS DEL SISTEMA

- **FUNCIONALIDAD: Que el software haga el trabajo para el que fue creado.**
- **CONFIABILIDAD: Que lo haga bien.**
- **DISPONIBILIDAD: Que todos los que quieran utilizar el sistema no tengan problemas.**
- **SEGURIDAD: La persona que no esté autorizada no debe tener acceso al sistema o a parte de él.**
- **INTEGRIDAD: Que la información sea verídica e igual en todos lados.**
- **ESTANDARIZACIÓN: Las características de la interfaz de usuario deben ser comunes entre múltiples aplicaciones.**
- **INTEGRACIÓN: Que todos lo módulos sean de fácil acceso.**

OBJETIVOS DEL SISTEMA

- **CONSISTENCIA: Que el apoyo visual sea igual en todas las pantallas (ej. ventanas similares). También se refiere a la terminología y los comandos usados en la interfaz.**

PORTABILIDAD: Que el paquete sea reconocido por la mayoría de los sistemas operativos.

CALENDARIZACIÓN: Respetar fechas límites para la culminación del proyecto.

PRESUPUESTO: No rebasar el presupuesto acordado.(el 70% del costo del software se gasta en el mantenimiento).

Diseño de diálogos

8 “guias de oro”

* ***Consistencia:***

  Debe ser congruente la terminología, y las guías seguidas, para evitar confusión al usuario. Si por ejemplo, en el mismo contexto apareciera en un lugar "archivo" y en otro "documento", esto causaría duda en el usuario, sobre el significado de los mismos.

* ***Atajos para usuarios frecuentes:***

  Los usuarios casuales pueden considerar buena la interfaz, pero cuando un usuario frecuentemente hace uso del diálogo, requerirá de herramientas para agilizar su trabajo; tales como el uso de atajos mediante el teclado (shortcuts, accelerators o hot-keys). *Recordemos que entre los objetivos de una buena interfaz están reducir el tiempo de respuesta y aumentar la productividad.*

* ***Retroalimentación:***

  Toda acción del usuario, debe mostrar sus efectos; esto para evitar que el usuario dude de que haya

  ocurrido la acción. Esto se logra mediante mensajes, iconos, cambios en la figura que representa

  el ratón, cambios en la pantalla, o sonido, entre otros métodos.

  Por ejemplo, siempre debe haber un indicador de avance, un reloj de arena o algún elemento similar, cuando se está realizando un proceso largo. De los dos elementos mencionados, será mejor un indicador de avance, pues le da mayor información al usuario y da expectativas apropiadas.

Diseño de diálogos

* ***Diálogos Cerrados:***

  Se dice que es cerrado y no abierto un diálogo, cuando le da una sensación de avance al usuario, tal que no siente que faltó algo dentro del diálogo; que el diálogo le dio toda la información necesaria al programa.

* ***Manejo simple de errores:***

  El usuario requiere la información suficiente para que un error tenga el menor impacto posible; muchas veces, al ocurrir un error por parte del usuario o del sistema, el usuario no tiene los elementos para recuperarse.

* ***Acciones reversibles:***

  Sobre todo los usuarios novatos, suelen tener miedo o ansiedad al enfrentarse a la interfaz. Uno de los elementos que permiten darle confianza al usuario, es saber que siempre tiene una forma de recuperar el trabajo que puede perder por un error.

* ***Sensación de control:***

  A veces el ordenador debe generar una acción; sin embargo, en la generalidad de las ocasiones, no debe ocurrir nada si no es porque el usuario se lo "pide" al ordenador. El usuario debe sentir que tiene control sobre el sistema, no sentirse manejado, para poder utilizar al máximo al sistema, y confiar en él para conseguir sus fines.

* ***Carga a la memoria de corto plazo:***

  La memoria de corto plazo es muy limitada; en promedio, podemos manejar sólo 7 elementos (items) a la vez. Si saturamos al usuario de información, y le requerimos que recuerde muchos elementos mientras trabaja, le hacemos más complicado el uso del sistema. El enfoque contrario es llevarlo paso a paso; si le dejamos manejar muy pocos elementos, se hace lento.

Manejo de errores

- Objetivo:
  - *Mejorar la productividad, reduciendo los costos debidos a errores.*
- Acciones:

  - *Mensajes de error claros y entendibles por cualquier usuario*

  - Ayuda en línea, sensitiva a contexto.

  - Prevenir errores.

Técnicas de prevención de errores

- Automatizar los procesos que causan fallas comunes
  - Ejemplo: cerrado de paréntesis
- Avisar de un estado inconsistente, cuando no es posible lo anterior.
  - Ejemplo: syntax error

Técnicas de prevención de errores

- Creación de macros (metacomandos)
  - *Ejemplos: si hay 7 pasos que requieren una secuencia estricta, usar un sólo botón o comando que englobe a los 7.*
- Corrección automática de comandos.
  - *Ejemplo: autocorrección de palabras en Word.*

Técnicas de prevención de errores

- Simplificación de comandos.
  - *Facilita al usuario el desempeño correcto (reduce la carga de aprendizaje*)
- Teclas con asociaciones fáciles a los comandos.

Guías para el despliegue de información

Objetivos: (Smith y Posier)

* *Consistencia. En terminología así como en formato.*
* *Asimilación eficiente de la información por parte del usuario.*
* *Minimizar la carga de la memoria del usuario.*
* *Compatibilidad entre el despliegue y la entrada.*
* *Flexibilidad para el usuario, en las formas de despliegue.*

Notas respecto a las guías

- Cada proyecto o aplicación debe establecer un manual o especificación de estándares, que incluye los formatos y terminología usados.
- Conseguir la atención del usuario.

Conseguir la atención

- La información importante o excepcional debe resaltarse:
  - *Intensidad ((hasta dos niveles)*
  - *Marcado*
  - *Subrayado*
  - *Encuadrado*
  - *Apuntador o carácter especial*
  - *Tamaño (hasta cuatro tamaños)*
  - *Fuente de texto (hasta tres)*
  - *Color (hasta 4 básicos)*
  - *Vídeo inverso y parpadeo*
  - *Sonido*

Guías para la entrada de datos

Objetivos *(Smith y Mosier)*

* *Consistencia entre los distintos puntos de entrada de datos*

* *Minimizar las acciones de entrada por parte del usuario*

* *Minimizar la carga de la memoria del usuario*

* *Compatibilidad entre el despliegue y la entrada*

* *Flexibilidad para el usuario, en las formas de entrada.*

Prototipos

Objetivo primordial:

* *Que el usuario experimente con la interfaz, aún sin funcionalidad*
* *Descubrir la reacción del usuario al sistema*

Características

* *Deben darse en una etapa temprana del ciclo de desarrollo.*

Pruebas de aceptación

Objetivos:

* *Comprobar los resultados de la interfaz*
* *Control de calidad*
* *Validación de teorías y guías*