

# Tema 6. Diseño de interfaces de usuario adaptadas

Formación específica, cursos verano 2008  
ETS de Informática Aplicada  
Universidad Politécnica de Valencia



# REGLAS BÁSICAS DE DISEÑO

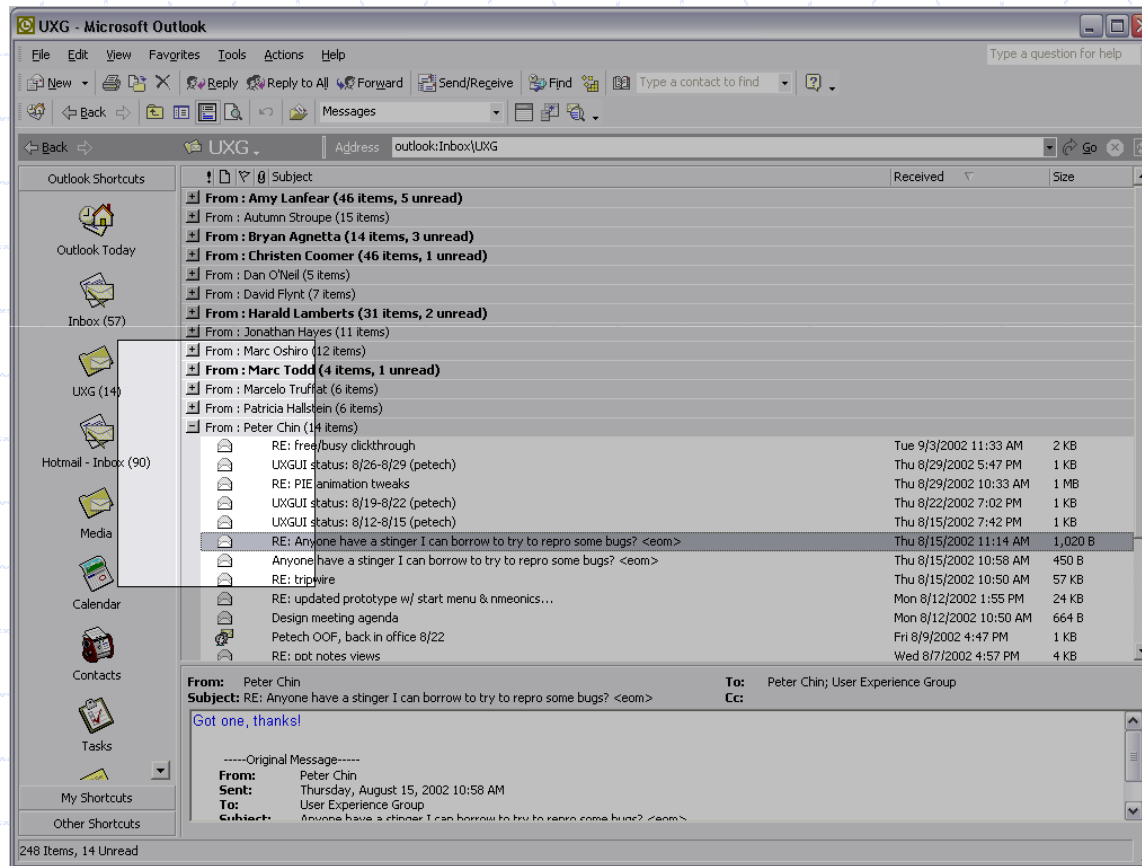
# La usabilidad lo primero

- ◆ La pantalla es pequeña
- ◆ No tenemos un teclado
- ◆ Ojo con la batería

# El tamaño de la pantalla

- ◆ Diseñar el UI para el tamaño de pantalla que tenemos
  - No incluir excesiva información en un formulario no se puede leer bien
  - Evitar en lo posible el uso del 'scroll', sobre todo del horizontal

# 1024x768 → 176x220



# No tenemos teclado

- ◆ Uso del SIP (Soft Input Panel), pero éste puede cubrir los controles en los que se utiliza
- ◆ En Pocket PC la pantalla es táctil, se puede usar el dedo.
- ◆ En SmartPhone no hay SIP
  - Cada tecla del teclado representa N caracteres
  - Usar TextBox sin más es alfanumérico
  - Cambiar a otro modo requiere varias pulsaciones o que el programador lo facilite

# Cuidado con la batería

- ◆ El mayor gasto de batería se va en el procesador
  - No hay elementos mecánicos como un disco
- ◆ No dejar aplicaciones con threads corriendo
  - Suele ser mejor esperar a que el usuario inicie la acción

# Otras consideraciones: Tamaño y forma

- ◆ Uso en ambientes concurridos y ruidosos
  - Órdenes por voz no adecuadas
- ◆ Interacción usuario-aplicación
  - Tipos de interacción
    - ◆ Con una mano (*"one-hand friendly"*)
    - ◆ Con dos manos (*"stylus friendly"*)
    - ◆ Con los dedos (*"finger friendly"*)
  - Desarrollo/prueba de aplicaciones en PC utilizando emuladores  
→ ¡Cuidado!
- ◆ Diseño de aplicaciones teniendo en cuenta limitaciones de batería y conexión

# Requisitos de fiabilidad

- ◆ La aplicaciones móviles deben estar disponibles 24h/día 7d/semana
  - Gestión eficaz de excepciones
    - ◆ Evitar fenómenos colaterales
  - Pruebas de uso de la interfaz por usuarios externos
    - ◆ Evitar bloqueos de la interfaz
  - Mecanismos para el almacenamiento persistente de la información local y remoto
    - ◆ Transparentes al usuario

# Productividad

- ◆ *"take it out of your pocket and use it right away"*
  - Interacciones de corta duración
  - Ejecución concurrente de aplicaciones
    - ◆ Notas/Agenda/Llamada
  - Tiempos de arranque/parada de aplicación reducidos
    - ◆ Modelos "always on"/"instant on"

# Principio acción/reacción

- ◆ Cuando el usuario realiza una acción espera una respuesta
  - Pulsar botón -> 2 seg. -> Volver a pulsar
  - Esencial: reconocer inmediatamente la petición del usuario mostrando
    - ◆ Resultado
    - ◆ Mensaje/Pantalla "Petición recibida y en procesamiento" → procesar nuevas peticiones del usuario
    - ◆ Mensaje/Pantalla "Petición recibida y en procesamiento" + Cursor de Espera/Barra de progresión → esperar el fin del procesamiento en curso

# Ejemplo

```
private void button_Click(object sender, EventArgs e)
{
    System.Threading.Thread.Sleep(4000);
}
```



!!! MAL !!!

# Ejemplo

```
private void button_Click(object sender, EventArgs e)
{
    System.Windows.Forms.Cursor.Current =
        System.Windows.Forms.Cursors.WaitCursor;

    System.Threading.Thread.Sleep(4000);

    System.Windows.Forms.Cursor.Current =
        System.Windows.Forms.Cursors.Default;
}
```



!!! Bien !!!

# Ejemplo

```
private void button_Click(object sender, EventArgs e)
{
    button3.Text = "Procesando";
    button3.Update();

    System.Windows.Forms.Cursor.Current =
        System.Windows.Forms.Cursors.WaitCursor;

    System.Threading.Thread.Sleep(2800);

    button3.Text = "Ya casi está";
    button3.Update();

    System.Threading.Thread.Sleep(1200);

    button3.Text = "button3";
    button3.Update();

    System.Windows.Forms.Cursor.Current =
        System.Windows.Forms.Cursors.Default;
}
```

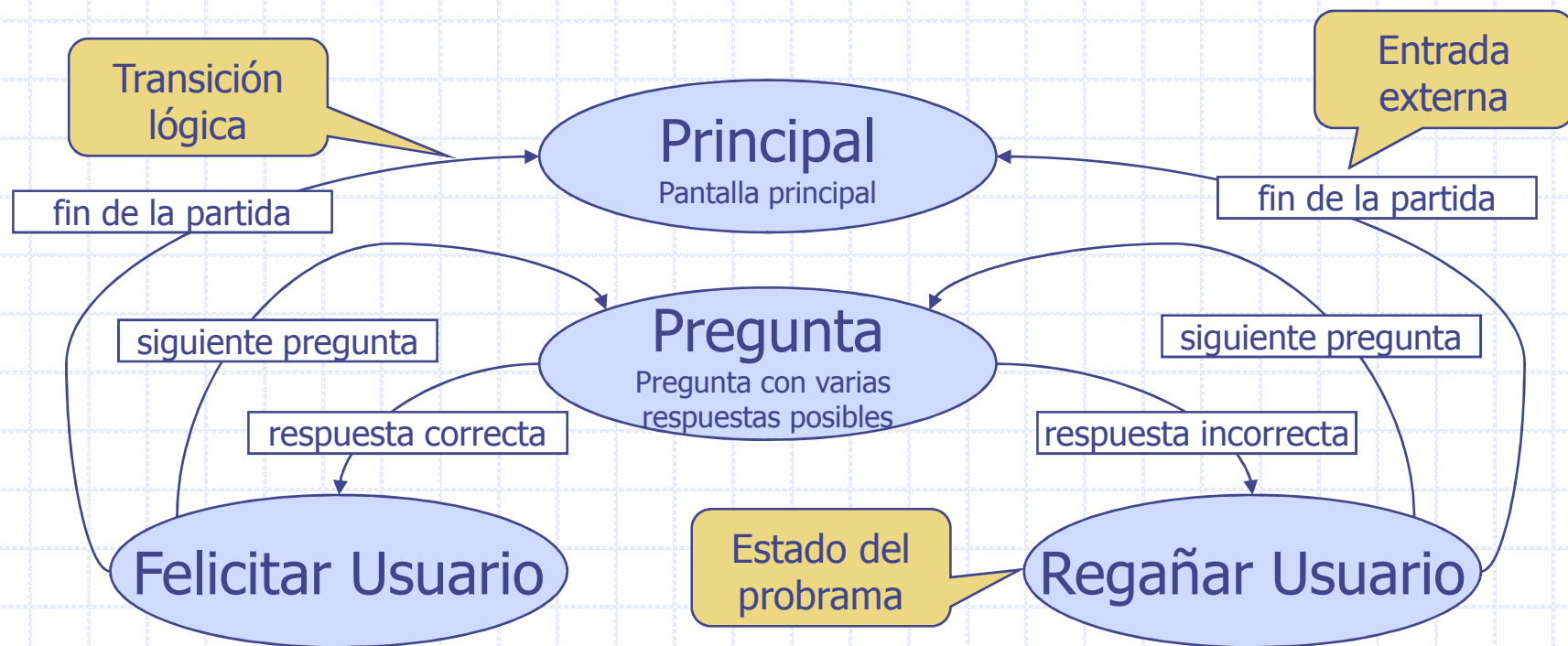


# Consistencia del GUI

- ◆ Los usuarios perciben las aplicaciones móviles como herramientas dentro de un entorno homogéneo
  - Seguir las normas de diseño de cada familia de dispositivos
    - ◆ Adaptación automática del GUI de la aplicación a la plataforma de ejecución
    - ◆ Desarrollo de GUIs para plataformas de ejecución específicas

# Diseño del GUI

- ◆ Recomendación: Uso de máquinas de estado para diseñar el GUI



```
Class MiMaquinaDeEstados{
  private enum EstadoJuego
  {
    Principal, Pregunta, RegagnarUsuario, FelicitarUsuario
  }

  private EstadoJuego estadoActual;

  private void CambioEstado(EstadoJuego nuevoEstado)
  {
    switch (nuevoEstado)
    {
      case Estado.Juego.Principal:
        if ((estadoActual != RegagnarUsuario) &&
            (estadoActual !=FelicitarUsuario))
        {
          throw new System.Exception("Transicion ilegal");
        }
        //Posicionar/actualizar los controles del formulario pantalla ppal
        //Muestra el formulario
        break;

        ...
      default:
        throw new System.Exception("Estado desconocido");
    }
    estadoActual = nuevoEstado;
  }
}
```

# Modificación implícita o explícita del estado de la aplicación

```

namespace EjemplosTema3
{
    public partial class Form2 : Form
    {
        private void button1_Click(object sender, EventArgs e)
        {
            textBox1.Visible = true;
            textBox2.Visible = false;
        }

        private void Form2_Load(object sender, EventArgs e)
        {
            textBox1.Visible = true;
            textBox2.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            textBox1.Visible = false;
            textBox2.Visible = true;
        }
    }
}

```

Mal diseño

```

namespace EjemplosTema3
{
    public partial class Form2 : Form
    {
        private void button1_Click(object sender, EventArgs e)
        {
            CambiaEstado(t1vt2i);
        }

        private void button2_Click(object sender, EventArgs e)
        {
            CambiaEstado(t1vt2i);
        }

        private void CambiaEstado(Estado ne)
        {
            switch(ne){
                case t1vt2i:
                    textBox1.Visible = true;
                    textBox2.Visible = false;
                    break;
                case t1it2v:
                    textBox1.Visible = false;
                    textBox2.Visible = true;
                    break;
            }
        }
    }
}

```

Buen diseño