El Zen y el arte del Software Libre

Conoce a los usuari@s, conocete a ti mism@.

Sep 30, 2005

24 slides

Enrico Zini (enrico@debian.org)

Enrico Zini enrico@debian.org

De que hablaré

- Introduciré dos conceptos: "situación" y "frustración"
- Explicaré como hacer:
  - Especificaciones sociales
  - Diseño social
  - Testing social
  - Debugging social

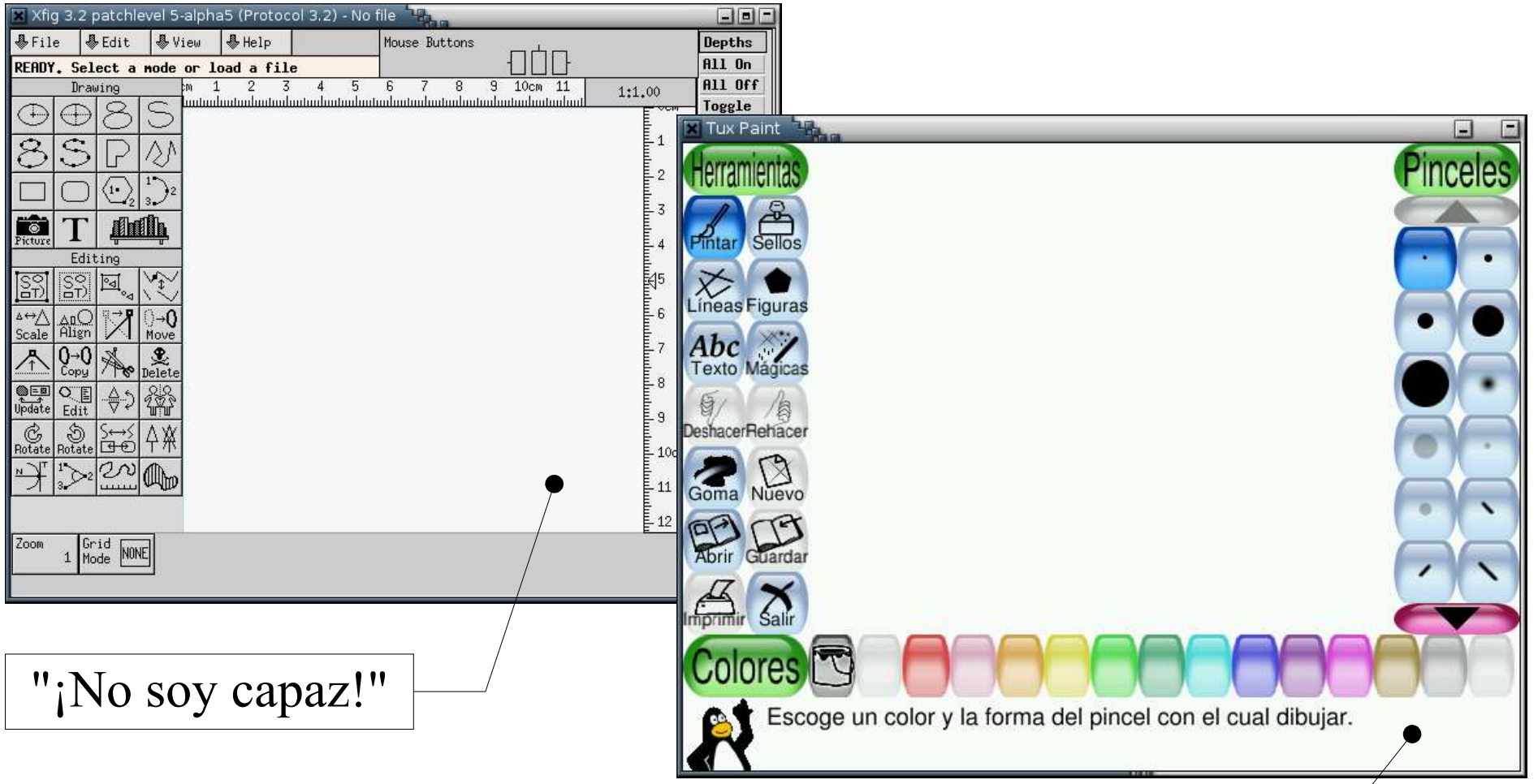
...todo esto en una hora: ¿list@s?

Enrico Zini enrico@debian.org

Escribir software es un acto creativo muy potente. Hagámoslo bien.

Enrico Zini enrico@debian.org

El software forma la identidad



"¡No soy capaz!"

"¡No puedo no ser capaz!"

Enrico Zini enrico@debian.org

El software forma la sociedad

Dos conceptos importantes.

La situación

¿Quien es usted, aquí y ahora?

¿Puedo formar la identidad de mis usuari@s? ¡Qué raro!

¿Puedo formar la identidad de mis usuari@s? ¡Qué idiotez!

¿Puedo formar la sociedad? ¡¡Revolución!!

¡Mira qué rico el morenito a mi derecha!

¡Mira qué rica la morenita a mi izquierda!

¡Mira qué español ridiculo el tio con pelo largo!

¿Cuando se come?

(pensar acciones situadas es como piensar a la vida con informaciones de runtime)

Enrico Zini enrico@debian.org

La frustración

s.f. Fracaso en el intento de obtener determinado resultado.

de "Clave, diccionario de uso del español actual"

(¿como hace frente usted a la frustración?)

Enrico Zini enrico@debian.org

Trabajar con l@s usuari@s

Trabajar con l@s usuari@s

Hay **usuari@s**.

Enrico Zini enrico@debian.org

Trabajar con l@s usuari@s

Hay **usuari@s**.

L@s usuari@s están **situados** en un contexto.

Enrico Zini enrico@debian.org

Trabajar con los usuarios

Hay **usuari@s**.

L@s usuari@s están **situados** en un contexto.

En ese contexto, ell@s tienen **metas**.

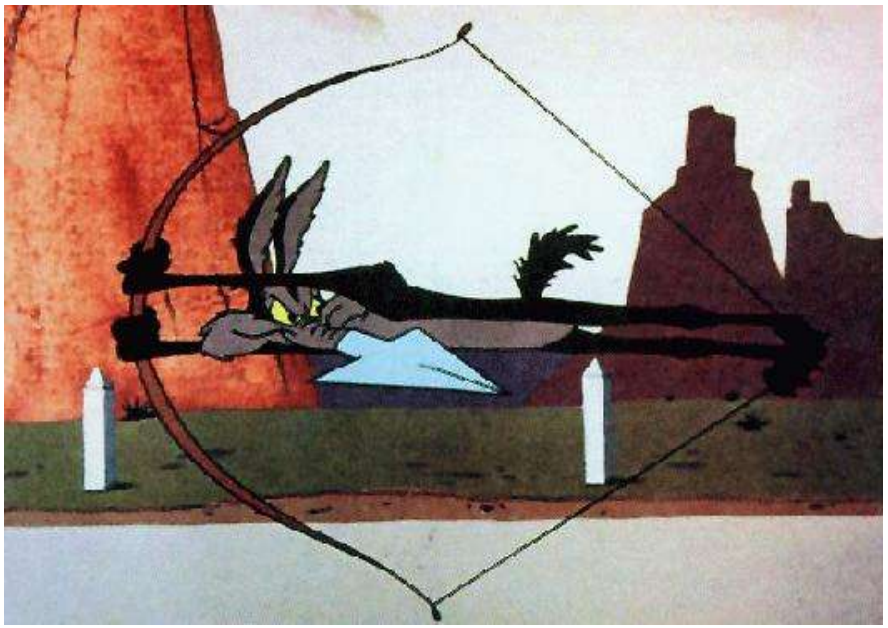
Trabajar con los usuarios

Hay **usuari@s**.

L@s usuari@s están **situados** en un contexto.

En ese contexto, ell@s tienen **metas**.

Para consiguir sus metas, necesitan **tareas**.



Trabajar con los usuarios

Hay **usuari@s**.

L@s usuari@s están **situados** en un contexto.

En ese contexto, ell@s tienen **metas**.

Para consiguir sus metas, necesitan **tareas**.

Si no se consigue nada, hay **frustración**.

Enrico Zini enrico@debian.org

Instrumentos que podemos utilizar en el diseño del software

Enrico Zini enrico@debian.org

Especificaciones sociales: usuari@s

¿**Para quien** estoy desarrollando?

Esto se puede especificar utilizando *"personas"* :

Persona: descripción en detalle del usuari@ medi@ ideal.

(¿es usted usuari@ de su software?
Ejercicio mental: cree una persona que describa si mism@)

Enrico Zini enrico@debian.org

Especificaciones sociales: metas

¿Que quieren l@s usuari@s de su software?

Sus usuari@s tienen *metas* (todos tenemos, de cualquier manera):

Personales: "no sentirse estupidos", "que la computadora haga la mayoria del trabajo"

De trabajo: "envíe el papel a la conferencia", "aumenten las ventas"

Practicas: "incorpore los dados", "busque una dirección en el directorio"

Falsas: "utilizar poca CPU", "que sea una aplicación web", "que sea fácil de utilizar"

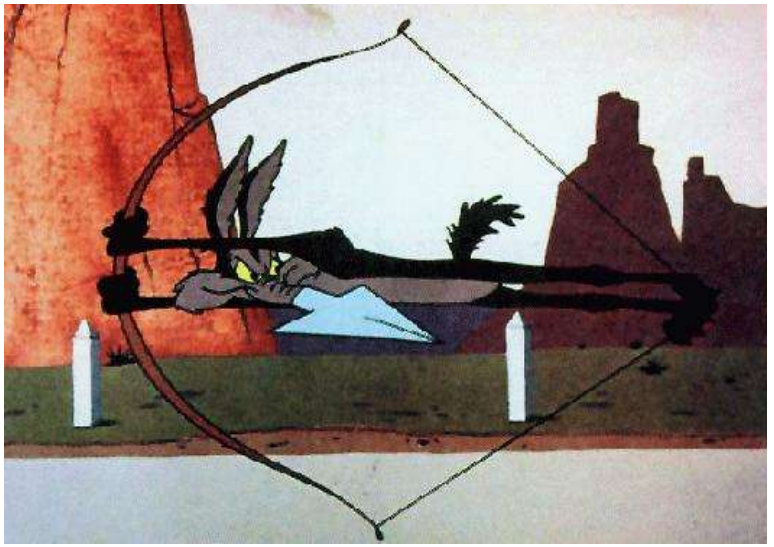
(Segundo ejercicio mental: cuales son sus metas en este talk?)

Enrico Zini enrico@debian.org

Design social: tareas

Que hacen sus usuari@s para conseguir sus metas?

- Tome la descripción de su *persona*
- *Situelas* en el ambiente apropiado
- Tome la lista de *metas* que tiene
- Cual es la mejor manera para que esa persona alcance sus metas con la cantidad mínima de *frustración*?



Bienvenid@s en el mundo del "*Task Analysis*"!

(con que tareas consiguen sus metas? Habría un diseño mejor?)

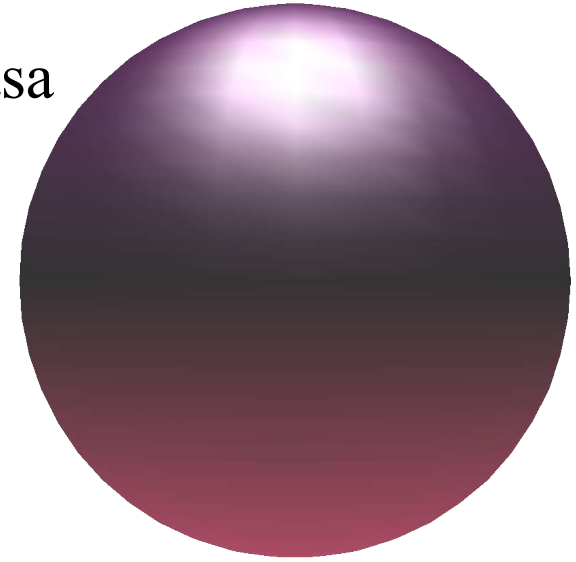
Enrico Zini enrico@debian.org

Diseño social: el numero mágico 7±2

Cache de trabajo: **7±2** cosas atomicas y arbitrarias

Query asociativas

Memoria de masa con capacidad virtualmente ilimitada



Organos externos: sentidos, musculación...

(búsquen el numero magico 7±2 en su vida diaria!)

Enrico Zini enrico@debian.org

Testing social

La tecnica de valutación eurística de Nielsen:

I. Visibility of system status
II. Match between system and the real world
III. User control and freedom
IV. Consistency and standards
V. Error prevention
VI. Recognition rather than recall
VII. Flexibility and efficiency of use
VIII. Aesthetic and minimalist design
IX. Help users recognize, diagnose, and recover from errors
X. Help and documentation

(prueben esa checklist en una interfaz que detesten!)

Enrico Zini enrico@debian.org

Debugging social

Tecnica de Flánagan de los Incidentes Criticos

Incidente critico: un comportamiento eficaz o ineficaz interesante.

Preguntas para debuggar l@s usuari@s:

- Describa qué condujo a la situación.
- Que hizo exactamente que era especialmente eficaz o ineficaz?
- Cual fue el resultado de esta acción?
- Porqué fue esta acción eficaz, o que acción mas eficaz pudo haber esperado?

(ahora saben que preguntar cuando le escriben "esta basura no funciona!")

Enrico Zini enrico@debian.org

Resumen

Recapitulación

El problema:

L@s **usuari@s** en una determinada **situación** tienen **metas** que consiguen hacendo **tareas**.

Si no consiguen sus metas, hay **frustración**.

Algunas soluciones:

Diseño de **personas** (usuari@ medio ideal).

Diseño de las **metas** (personales, de trabajo, practicas, falsas).

Diseño de las **tareas**
(como puede esta persona en este contexto conseguir esto?).

Evaluar los limites de l@s usuari@s (**¡siete màs o menos dos**!)

Evaluar las interfaces (checklist como la **Euristica de Nielsen**)

Evaluar los problemas (preguntas para los **Incidentes Criticos**)

Enrico Zini enrico@debian.org

Conclusión

Muchos proyectos de software no se interesan mucho por sus usuari@s: ustedes han visto que se les puede comprender, y hacer felices.

Saber como comprender a sus usuari@s y hacerl@s felices es muy importante: usted podria estar entre ellos!

Enrico Zini enrico@debian.org