

Diseño de Interfaces de Usuario

Análisis de Usuario y Actividades

Interfaces Pessoa Máquina

Marcos Ortega (m.ortega@udc.es)

Análisis de Usuario y Actividades

- ¿Por dónde empezamos el desarrollo de IU?
- ¿Cómo obtenemos información para nuestros primeros diseños?
- Primera etapa del proceso centrado en usuario (UCD):
 - ¿Quiénes son los usuarios? (usuario)
 - ¿Qué necesitan hacer? (actividades)
- Aproximación basada principalmente en actividades o tareas: Task-centered design

UCD: Conocer al usuario

- Identificar características del conjunto objetivo de usuarios
 - Educación
 - Capacidades físicas
 - Experiencia en informática en general
 - Habilidades de tecleo, lectura...
 - Experiencia en el dominio
 - Experiencia sobre la aplicación
 - Entorno de trabajo, contexto social
 - Patrones de comunicación

UCD: Múltiples usuarios

- Muchas aplicaciones tienen distintos tipos de usuarios
- Aplicación de un centro clínico para gestionar admisiones, diagnósticos, historiales, tratamientos, cobros
 - Recepcionistas
 - Médicos
 - Contables
 - ...

UCD: Cómo conocer al usuario

- Obstáculos
 - Los desarrolladores y los usuarios están sistemáticamente separados con frecuencia
 - Soporte técnico aísla a los desarrolladores de los usuarios
 - Ventas/Marketing aísla a los usuarios de los desarrolladores
 - Algunos usuarios son muy difíciles de llegar
 - Directivos, médicos...

UCD: Conocer las necesidades reales

- Conocemos quiénes son los usuarios pero no sabemos qué necesitan (ellos tampoco con precisión)

AVERIGUAR PROBLEMAS	AVERIGUAR SOLUCIONES
¿Qué es lo que hace la gente actualmente?	Encontrar una manera de hacerlo más fácil y eficiente
¿Qué tienen que hacer y no les gusta?	Conseguir que no tengan que hacerlo o, al menos, más entretenido
¿Qué les gustaría hacer?	Hacerlo posible

UCD: Técnicas de investigación

- Técnicas para averiguar lo que hacen los usuarios
 - Observación en el trabajo (*Job Shadowing*)
 - Entrevistas en su contexto (*Contextual Interviews*)
- Los usuarios pueden ser muy diferentes a lo que tenemos en mente
- Requisitos
 - Tenemos que tener una idea previa de qué usuarios son
 - Accesibles

Job Shadowing

- Recordamos: los usuarios a veces no saben lo que quieren o no lo expresan de forma adecuada para los desarrolladores
- La idea del *job shadowing* es poder visitar a los usuarios en su entorno y poder extraer directamente información útil
 - Podemos extraer mucha información útil de la que el usuario no es consciente o no valora

Job Shadowing

- ¿Hay tareas específicas que requieren mucho tiempo?
- ¿Hace el usuario las mismas cosas repetidamente?
- ¿Ejecuta tareas con *workarounds*?
- ¿Requiere memorizar pasos o ciertos aspectos técnicos que podríamos gestionar automáticamente en un ordenador?
- ¿Usa herramientas externas tales como libreta, calculadora,...?

Entrevista en su contexto

- *Job shadowing* cumple la máxima: “Lo que ves es más relevante que lo que te cuentan”
- Sin embargo, mucho conocimiento del dominio solo puede extraerse hablando con los expertos
- Tras el proceso de observación, es conveniente pasar un tiempo con el usuario preguntándole sobre aspectos observados y otros aspectos que quizás no lo fueron

Entrevista en su contexto

- Queremos indagar en las posibles mejoras
- Evitar
 - Opiniones personales
 - Forzar al usuario a diseñar por nosotros
- Algunas cuestiones tipo:
 - ¿Hay tareas habituales que no haya visto hoy?
 - ¿Qué clase de problemas solucionas más a menudo?
 - ¿Por qué has realizado [tarea observada] de esa manera?
 - ¿Cómo hacéis si no hay suficiente información para completar una tarea?
 - ¿Con quién interactúas habitualmente en estas tareas?

Observación Remota (*Remote Shadowing*)

- A veces no es posible visitar *in-situ*
- Observación remota requiere colaboración
 - Grabaciones de escritorio
 - Grabación via cámara de vídeo
- No es necesario grabaciones de mucha calidad ni framerate
- Una ventaja es que podemos revisar los vídeos cuando sea necesario durante el proceso

Limitaciones de las técnicas

- Los humanos somos capaces de predecir situaciones o escenarios hipotéticos/futuros
- Desafortunadamente, no lo hacemos perfectamente
 - Los usuarios pueden no predecir correctamente cómo usarán un producto
 - Nosotros tampoco podemos predecir muy bien cómo lo harán
- La conclusión es: nunca podremos llegar al 100% de seguridad
 - Hay que saber cuándo parar
 - Preferible pasar a siguiente etapa y evaluarla

Personas

- *Personas* (usado en inglés): son representaciones ficticias de ciertos grupos objetivo
- Útiles para poder resumir información en una persona que en un “segmento”
- Previo a su creación necesitamos haber investigado sobre nuestros usuarios

Personas

- A través de investigación de usuarios sabemos
 - Problemas a resolver
 - Clase de usuarios que se benefician
- Esta información se usará muy habitualmente
- La representación en *personas* nos ayuda a comunicarnos de forma más sencilla
- Ayuda a focalizar el producto identificando categorías o tipologías entre la audiencia de usuarios
- Ayuda a hacer el diseño más centrado en usuarios

Personas

- Limitaciones

- Son representaciones imaginarias: no te van a llevar la contraria
- No evita el contacto directo con usuarios en todas las etapas de desarrollo
- Lleva tiempo hacerlo bien: hay que procesar mucha información y sintetizar aquella que es útil
- En un equipo de trabajo pequeño no aporta grandes ventajas
 - Más o menos todo el mundo tendrá una idea correcta de los usuarios involucrados

Personas

- Cómo crear personas
- Hay que empezar con observación y entrevistas
 - Podemos empezar pensando que hay numerosas tipologías de usuarios y acabar reduciendo este número sensiblemente
- Agrupar lo más posible
 - Habitualmente tres o cuatro personas como mucho son suficientes
 - Los objetivos deben estar muy bien definidos


Personas


- Es importante añadir detalles relevantes
 - Nivel de habilidad en ciertas tareas
 - Edad / Experiencia
 - Importancia
 - Dispositivos que usa (p.ej. si accede a la web via móvil...)
- También se recomienda añadir detalles irrelevantes
 - Ayuda a recordarlas
 - Evita demasiada elasticidad
 - Algunos detalles observados irrelevantes puede que no lo sean tanto en siguientes etapas

Personas

- Típicamente se les da un nombre y una imagen o avatar
- El nombre no debería reflejar nadie *real*
- Son un mecanismo interno de comunicación y diseño
- Aunque nuestras *personas* sepan manejar muy bien los ordenadores, todavía necesitamos evaluar nuestros diseños con usuarios reales

Personas

	<p>Persona Name:</p> <p>Job/Role Description:</p>
<p>Short Narrative (description of the persona acting out his or her primary scenario(s)):</p>	
<p>Data Sources and/or Sources of Assumptions:</p>	

	<p>Persona Name:</p> <p>User Class or Segment (including market size, importance):</p>
<p>Job, Role, Activities:</p>	
<p>Goals:</p>	
<p>Abilities, Skills, Knowledge:</p>	
<p>Personal Details:</p>	
<p>Data Sources and/or Sources of Assumptions:</p>	

Diseño centrado en actividades (ACD)

- El diseño centrado en usuarios es muy útil y recomendable
- Problema: no siempre es posible o conveniente
- A veces es mejor centrarse en las actividades (o tareas) que se van a llevar a cabo en nuestra aplicación
- Es una aproximación intuitiva más clásica
 - Y la más habitual y abundante: webs, el volante de un coche, el pomo de una puerta...
- Si el producto está orientado a una audiencia general, la investigación sobre el usuario es de poca ayuda en una etapa inicial

Diseño centrado en actividades vs usuarios

- En cualquier caso, es importante decidirlo lo antes posible para enfocar las tareas de análisis y recolección de información de forma adecuada
- Es importante hacer que tu producto se adapte a los usuarios de forma fácil
 - No siempre hay que adaptar el producto a una serie de usuarios

Analizar Tareas

- Identificar tareas individuales que el programa debe hacer
- Cada tarea es un objetivo (*qué* y no *cómo*)
- A menudo es conveniente empezar con unos objetivos globales y posteriormente refinarlos en subobjetivos
- Más cercano al análisis en ingeniería de software: casos de uso, historias de usuario,...

Partes esenciales en el Análisis de Tareas

- ¿Qué hay que hacer?
 - Objetivos
- ¿Qué es necesario previamente?
 - Precondiciones
 - Tareas que condicionan la actual
 - Información que depende del usuario
- ¿Qué pasos están involucrados?
 - Subtareas
 - Proceso recursivo

Análisis de Tareas

- Otras preguntas
 - ¿Dónde se realiza esta tarea? Contexto, escenario
 - Frecuencia de la tarea
 - Restricciones temporales o de otro tipo
 - ¿Cómo se aprende la tarea?
 - ¿Qué puede salir mal?
 - ¿Quién más está involucrado en la tarea?

Ejemplo: Checkout de autoservicio en supermercado

- Queremos diseñar un sistema de *checkout* auto-servicio en un supermercado
- Usuarios de muy amplio espectro
- No podemos asumir demasiados conocimientos previos por su parte
- Consideramos dos perfiles de usuarios
 - Comprador
 - Asistente de la tienda

Ejemplo: Checkout de autoservicio en supermercado

Una posible tarea:

- Objetivo
 - Pasar los productos por la registradora
- Precondiciones
 - Todos los productos ya están en el carrito
- Subtareas
 - Pasar productos pre-empaquetados
 - Pasar productos sueltos

Ejemplo: Checkout de autoservicio en supermercado

- ¿Dónde se realiza esta tarea?
 - Delante de las cajas habilitadas, de pie
- Frecuencia de la tarea
 - Una/dos veces a la semana
- Restricciones temporales o de otro tipo
 - Un par de minutos
- ¿Cómo se aprende la tarea?
 - Por observación de otros clientes, intentándolo o siendo atendido por un asistente
- ¿Qué puede salir mal?
 - Código de barras defectuoso o no presente
 - Edad inadecuada para ciertos productos

Jerarquías en el producto

- Conocemos actividades, necesidades de los usuarios
- Normalmente toda esa información la querremos agrupar en jerarquías
 - Facilita la comprensión a los usuarios
- Jerarquías están presentes en numerosas aplicaciones de forma explícita: p.ej. navegadores web
 - Ventana (Nivel 1)
 - Pestaña (Nivel 2)
 - Barra/Botones (Nivel 3)
 - Página web (Nivel 4)
- Normalmente el usuario espera que efectos invocados en un nivel solo afecten a niveles inferiores

Estructurar nuestro producto

- Técnica: Card Sorting
- Nos permite obtener información sobre lo que el usuario piensa de las partes individuales de nuestro producto
- Particularmente útil dependiendo de la complejidad del producto
 - Típicamente, una aplicación web con numerosas páginas entre las que podemos navegar para acceder a funcionalidad variada

Card Sorting

- Consideremos una web de fabricante de electrodomésticos
- Si tenemos una avería en nuestra lavadora y accedemos a la web en busca de info
 - Unos buscarán la sección de soporte
 - Otros irán a Productos > Electrodomésticos > Lavadoras > Servicio Técnico
- ¿Cómo sabes qué ruta será la más habitual?
- ¿Cómo sabes dónde espera típicamente la gente encontrar cierta información dentro de la jerarquía?

Card Sorting

- La idea es utilizar etiquetas de papel para representar los distintos índices o elementos de la estructura de la aplicación
 - En un caso de sitio web cada carta puede ser una página, o áreas del sitio web
 - En una aplicación gráfica podemos usar propiedades, items de menú, comandos, tareas, elementos visibles.
 - Depende también de lo refinado que tengamos ya el producto
- No conviene mezclar demasiado elementos muy dispares en la jerarquía

Card Sorting

- Los participantes pueden ser un único individuo o varios a la vez
 - En caso de varios además de dificultad de planificación normalmente acaba siendo dominado por un único usuario
 - Por otra parte, puede crear conversación, y facilita la extracción de información útil o de posibles alternativas
- Es preferible (aunque no siempre posible) hacer más ordenaciones pequeñas que pocas grandes

Card Sorting

- Conviene introducir correctamente el proceso y su justificación a los participantes
- Importante aclarar que se pretende que agrupen cartas similares pero desde el punto de vista de la aplicación (cosas que deberían ir juntas, pantallas accesibles una desde otra, etc)
- Repasar todas las cartas (elementos a agrupar, ordenar)
- Tomar notas es importante
 - El resultado es informativo pero también el proceso

Card Sorting

- Evaluar los resultados
- Conjunto cerrado vs abierto
- Conjunto cerrado
 - Conteo de en qué grupo es más frecuente un elemento
- Conjunto abierto
 - Identificar grupos (pueden ser palabras distintas pero representar lo mismo)
 - Si hay mucha disparidad en la forma de hacerlo, quizás haya que ofrecer varios caminos

Guías generales para jerarquías

- Permitir que las cosas existan en varios sitios
 - No hay que ser estricto para evitar pensamientos únicos
- ¿Superficial o profundo?
 - Superficial: se puede llegar a todo más rápido con menos clicks
 - Generalmente se considera una métrica positiva
 - No está directamente ligada a la satisfacción ni tasa de éxito
 - Minimizar el número de posibilidades al usuario también puede ser muy importante

Guías generales para jerarquías

- Agrupar elementos
 - Visualmente es más fácil y cómodo si el usuario percibe ciertos elementos estructurados
 - Por ejemplo, agrupar ciertas opciones en bloques comunes por tipo de funcionalidad permite descartar la mayoría de ellas de un plumazo
 - Según el contexto podemos hacer uso de distintas propiedades visuales: proximidad, color, tamaño, orientación, contenedores...

Modelo Mental

- Un usuario tiene ideas preconcebidas de cómo deben funcionar ciertas cosas
- El concepto que el usuario se forma de cómo funciona algo se llama *modelo mental*
- Normalmente es más sencillo que la realidad subyacente: Pedal del acelerador
- Que el modelo mental sea sencillo (o incluso equivocado) no impide que sea útil al usuario a saber manejar un sistema

Modelo Mental

- Nuestro producto típicamente refleja tres modelos diferentes:
 - Cómo el usuario cree que nuestro sistema funciona. Modelo Mental.
 - Cómo es presentado el producto al usuario. Modelo de diseño o modelo de IU.
 - Cómo se implementa el producto. Modelo de implementación o de programador

Modelo Mental

- En un producto ideal, los tres modelos son consistentes. El interfaz representa perfectamente la implementación y el usuario entiende perfectamente lo que ve.
- La realidad: nunca son consistentes
- Compromiso entre simplificar el interfaz y ser lo más consistente posible con la implementación

Modelo Mental

- Sistema de compra de películas online
- Modelo mental de compra de películas:
 - Ir a la tienda
 - Bucear entre una serie de películas
 - Intercambiar dinero por la película elegida
- En online: no hay dinero físico, hay una descarga involucrada, probablemente haya que descifrar o quitar algún tipo de marca del vídeo...
- El modelo de IU debe ser lo más cercano posible al modelo mental, tratando de esconder la complejidad de los detalles de implementación.

Modelo Mental: 7 principios

- Principios para ayudar al usuario a formarse modelos mentales válidos
 - Simplicidad
 - Familiaridad
 - Reconocimiento
 - Flexibilidad
 - Feedback
 - Seguridad
 - *Affordances* (potencial, adecuación)