Human-computer interaction

anno accademico 2018-2019

Nicola Rizzo - nicola.rizzo@supsi.ch

Interazione

- Lo scopo di un sistema interattivo è quello permettere a un utente di raggiungere degli obiettivi (*goals*) in un determinato dominio applicativo (*domain*)

  - dominio - definisce un'area di competenza e l'insieme di concetti importanti per modellizzarla

  - intenzione (*intention*) - azione specifica richiesta per raggiungere l'obiettivo

  - attività (*task*) - operazione che consente all'utente di manipolare i concetti di un dominio

  - obiettivo - è l'output desiderato da un task

Execution-evaluation cycle (Norman)

1. Stabilire il goal
2. Formulare l’intenzione
3. Specificare la sequenza di azioni
4. Eseguire l’azione
5. Percepire lo stato di sistema
6. Interpretare lo stato di sistema
7. Valutare lo stato di sistema rispetto agli obiettivi e alle intenzioni

Problemi di utilizzo, cause (Norman)

- Utente e Sistema si esprimono in linguaggi diversi:
    - Task language per l'Utente: descrive attributi psicologici del dominio rilevanti per lo stato dell'Utente
    - Core language per il Sistema: descrive attributi computazionali rilevanti per lo stato del Sistema
- Golfo d'esecuzione
    - Differenza fra la formulazione dell'utente delle azioni e quelle che il sistema consente di eseguire (l'Interfaccia dovrebbe cercare di minimizzarlo)
- Golfo di valutazione
    - Differenza fra la rappresentazione fisica dello stato del sistema e l'aspettativa dell'utente (se l'utente riesce a valutare velocemente quello che è successo, il golfo di valutazione è piccolo)

Errori dell’utente

- Slips - riguardano il golfo di esecuzione, per esempio l’utente clicca sul bottone sbagliato (vicino a quello voluto)

- Mistakes - sul golfo di valutazione; un utente che clicca sul bottone della lente pensando che sia “cerca”, mentre il sistema cambia livello di zoom

Errori dell’utente - soluzioni

- Slips - è possibile diminuirli anche solo cambiando di poco il design l’interfaccia (per esempio con maggiore spazio fra i bottoni)

- Mistakes - probabilmente parte dell’interfaccia deve essere riprogettata

Legge di Jakob (sulla Internet User Experience)

- Gli utenti passano la maggior parte del loro tempo su altri siti. Questo significa che preferirebbero che il tuo sito funzionasse nel modo che già conoscono.

8 regole d’oro sul design delle interfacce (Shneiderman)

1. Cercare d’essere consistente
2. Dare la possibilità agli utenti abituali di usare scorciatoie
3. Offrire feedback informativo
4. Progettare le conversazioni in modo tale che arrivino a una conclusione
5. Offrire gestione e prevenzione degli errori
6. Permettere una facile reversibilità delle azioni
7. Offrire un punto centrale di controllo
8. Ridurre il carico della memoria a breve

7 principi dell'Interaction Design (secondo Norman)

1. *Usare sia la conoscenza del mondo reale che quella mentale* in modo da rendere facile l'utilizzo dell'interfaccia con le informazioni visibili, ma anche in modo che siano facili da memorizzare

2. *Semplificare la struttura dei task* per evitare eccessivo "problem solving" e sovraccaricare la memoria dell'utente

3. *Rendere le cose visibili* golfi di esecuzione e di valutazione: l'effetto di un'azione sul sistema deve essere chiaro

4. *Costruire con chiarezza le associazioni* in modo che le intenzioni dell'utente si mappino correttamente su azioni del sistema

5. *Sfruttare il potere dei vincoli* sia naturali che artificiali. Devono essere permesse solo le azioni corrette, nel modo corretto (come per risolvere un puzzle)

6. *Progettare tenendo a mente gli errori* L'interfaccia deve cercare di anticipare gli errori umani e permettere facilmente di ripristinare il sistema

7. *Quando qualsiasi altra cosa non funziona, usa gli standard (aka non reinventare la ruota)* In modo che l'utente possa abituarsi a eseguire nuove associazioni

Complessità

- A volte la complessità fa parte del dominio e non è pensabile di rimuoverla o ridurla dall’interfaccia del sistema

Stili di interazione

- Riga di comando (CLI)
- Interfacce a menu
- Linguaggio naturale
- domanda / risposta e dialoghi a query (i motori di ricerca ne sono una specializzazione)
- Form e fogli di calcolo
- WIMP (Windows, Icons, Menus, Pointer) e sue varianti
- Punta e clicca (browser, certi videogiochi)
- Interfacce 3D

Riga di comando

- Adatta ai “power users”
- Adatte per accedere a sistemi remoti
- Curva d’apprendimento ripida

Interfacce a menu

- Punta più al riconoscimento che al ricordo (quindi richiede meno sforzo rispetto alla CLI per esempio)

- Le voci di menu devono essere ordinate e raggruppate logicamente per aiutare il riconoscimento

- Il raggruppamento talvolta impone una gerarchia, quindi non tutte le voci possono essere raggiungibili al primo livello

Linguaggio naturale

- Può essere fraintendibile (linguaggio naturale ambiguo)
- Se basato su grammatiche: difficile ricordare i termini da usare
- Linguaggio libero: possibili problemi di privacy

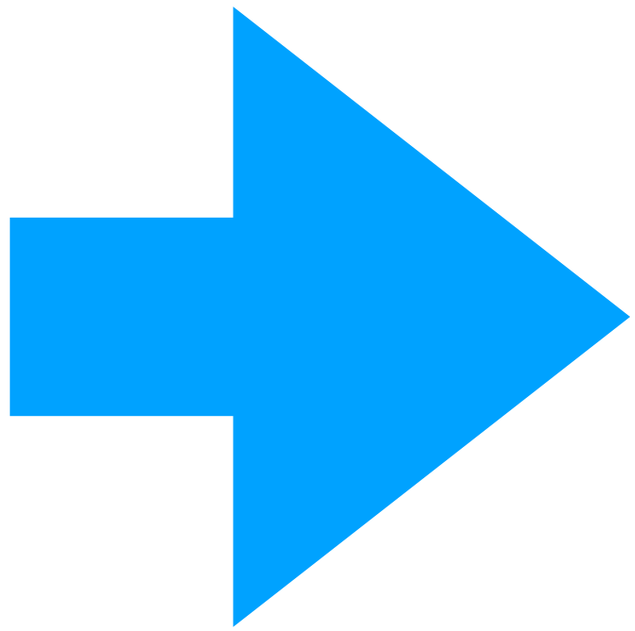
Domanda / risposta - Query

- Facilità d’apprendimento (se a risposta chiusa)
- Funzionalità limitata
- Difficoltà nel caso di linguaggi specifici (eg SQL)

Form e fogli di calcolo

- Facilità d’apprendimento (form)
- Necessitano di validazione
- Input e output sono rappresentati allo stesso modo, rendendo l’interfaccia flessibile (fogli di calcolo)

WIMP



Punta e clicca

Punta e clicca

- È il paradigma usato nelle pagine web
- Strettamente connesso con le interfacce WIMP
- Utilizzato in alcuni videogiochi

Interfacce 3D

- Elementi di 3D sono (stati) usati anche nelle interfacce WIMP
- Comprendono videogames, interfacce AR, VR
- Non sono sempre facili da usare

Elementi delle interfacce WIMP

- Finestre
- Icone
- Menu
- Puntatori
- (bottoni, toolbar, palette, dialog boxes…)

Legge di Fitts

$$t = a + b log_2(\frac{d}{s} + 1)$$

t = tempo
d = distanza
s = dimensioni
a, b costanti arbitrarie

Legge di Fitts

- Il tempo necessario ad acquisire un bersaglio è funzione della distanza e delle dimensioni del bersaglio

Legge di Hick

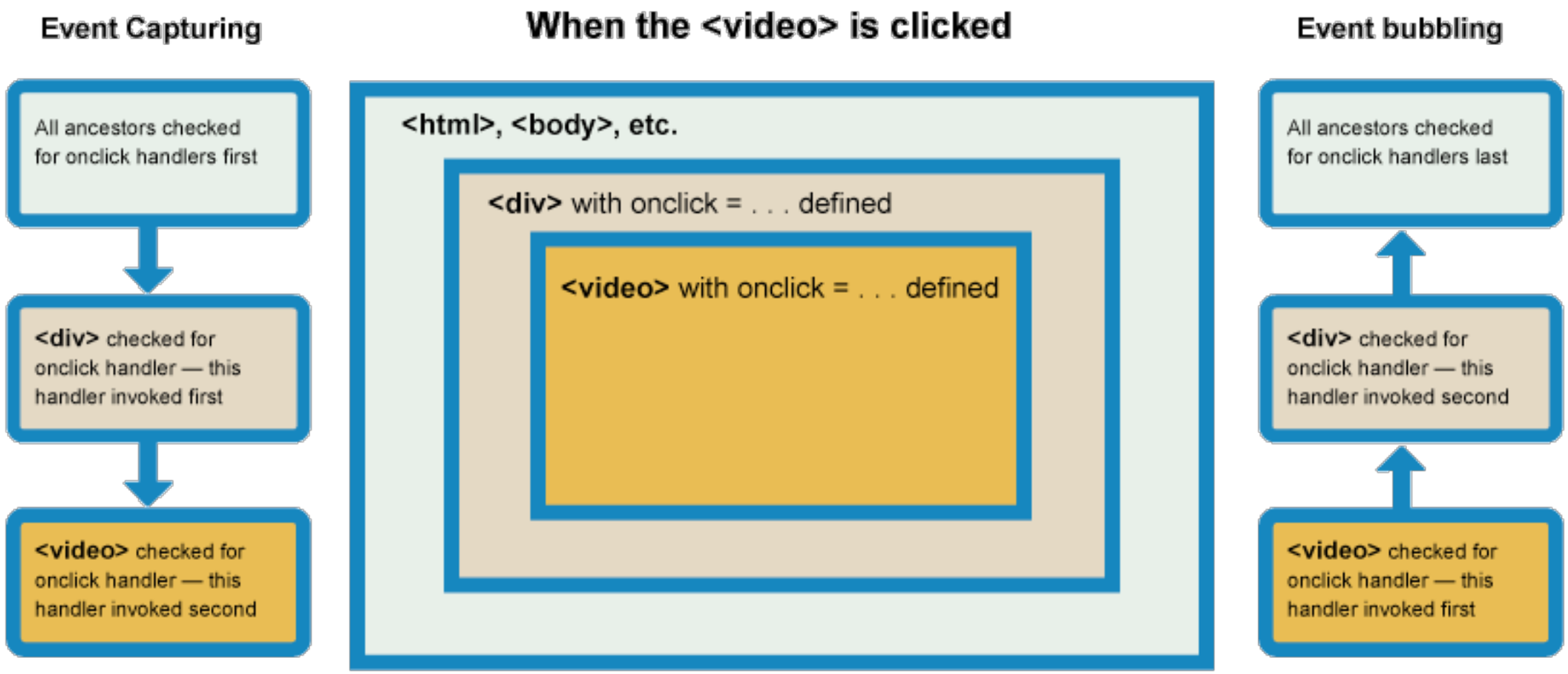
- Il tempo necessario a prendere una decisione cresce con il numero e la complessità delle scelte

Eventi

- Rappresentano l’accadere di qualcosa di interesse per l’applicazione

- Alcuni sono generati dall’utente, altri da API: sono fondamentali per gestire l’interazione

Eventi - propagazione



Event capturing

- In questa prima fase l’evento si propaga dagli elementi parent a tutti i discendenti. Per default questa fase non avviene

Event bubbling

- In questa fase l'evento si propaga dal target verso i nodi esterni.

- La fase di bubbling consente la tecnica dell'event delegation, che permette di aggiungere lo stesso listener a più elementi che condividono un antenato

Eventi - aggiunta di listener - very old school

```
…<div onevent=“doSomething()”></div>
```

Forma da non usare mai. Riportata solo per completezza, si basa sull’uso di variabili globali*

*In alcune librerie moderne questa forma è ammissibile e non conduce all’utilizzo di globals

Eventi - aggiunta di listener - old school

```
target.onclick = handler;

// dove handler è una funzione
```

Forma da usare solo se costretti. Da usare se è necessario il supporto di IE 9 o precedenti

Eventi - aggiunta di listener

```
target.addEventListener(tipo, listener[, opzioni]);
```

```
target.addEventListener(tipo, listener[, useCapture]);
```

la prima sintassi è per browser moderni (IE > 11), la seconda funziona su tutti (IE > 9)

Eventi - rimozione di listener

```
target.removeEventListener(tipo, listener[, opzioni]);

target.removeEventListener(tipo, listener[,
useCapture]);
```

la funzione listener e il parametro capture in opzioni o useCapture devono essere gli stessi* della chiamata addEventListener

*= medesimi: in particolare la funzione e l'oggetto devono essere la stessa istanza

Bibliografia

- Donald A. Norman, “Vivere con la complessità”