Criteri di Usabilità dei sistemi software

(Usabilità sistemi HMI)

Definizione minimalista di Usabilità

Con il termine **usabilità** di un software (o di qualsiasi oggetto prodotto dall'uomo per uno scopo specifico) si intende la **facilità** con cui una persona svolge un **compito** coerente con le funzioni dello stesso.

Questa definizione è eccessivamente semplice, perché il concetto di facile/difficile è assai soggettivo e perché essa non tiene conto delle conoscenze, abilità, intuizioni e influenze esterne che hanno portato a concepire, realizzare e distribuire il software in oggetto.

Infatti chi usa il software e lo ritiene difficile da usare, quindi poco usabile, spesso non è a conoscenza né dei vincoli e restrizioni legati al suo uso o del suo uso improprio. Il programma potrebbe essere adibito ad una funzione diversa da quella che l'utente tenta di svolgere.
Semplificando, se un software non risulta usabile allora o è colpa del progettista che ha sviluppato un programma non adatto al compito, oppure è colpa dell'utente che non conosce il vero compito per il quale il software è stato concepito.

Sebbene sia un argomento controverso, esistono definizioni condivisa sulle caratteristiche e funzioni che un software deve possedere per poter raggiungere un buon grado di usabilità. E' comunque preferibile considerare tipologie di software (come faremo nel seguito per le interfacce HMI) e applicare su di esse le regole, metodologie e linee guida più adatte.

Infine è importante distinguere usabilità da utilità, che mira alla soddisfazione di bisogni immediati e concreti, e da prodotti e oggetti culturali, per i quali lo scopo primario è dare piacere, soddisfazione, emozione, e non certo risolvere un compito.

Interfacce software

Il software è un prodotto con particolari caratteristiche. Innanzitutto non è un oggetto culturale, non produce sensazioni ed emozioni in quanto tale, non è materiale. Si evidenzia la sua presenza solo attraverso un altro componente, il computer, che gli consente di risolvere una specifica classe di problemi.

Passati i tempi in cui il progettista/sviluppatore di software era anche l'utente finale, e ormai consolidata la separazione, tra chi progetta il software e chi lo usa, si pongono concretamente tutti i problemi legati all'usabilità del prodotto.

Secondo R. Norman l'usabilità di un prodotto software misura la distanza cognitiva tra:

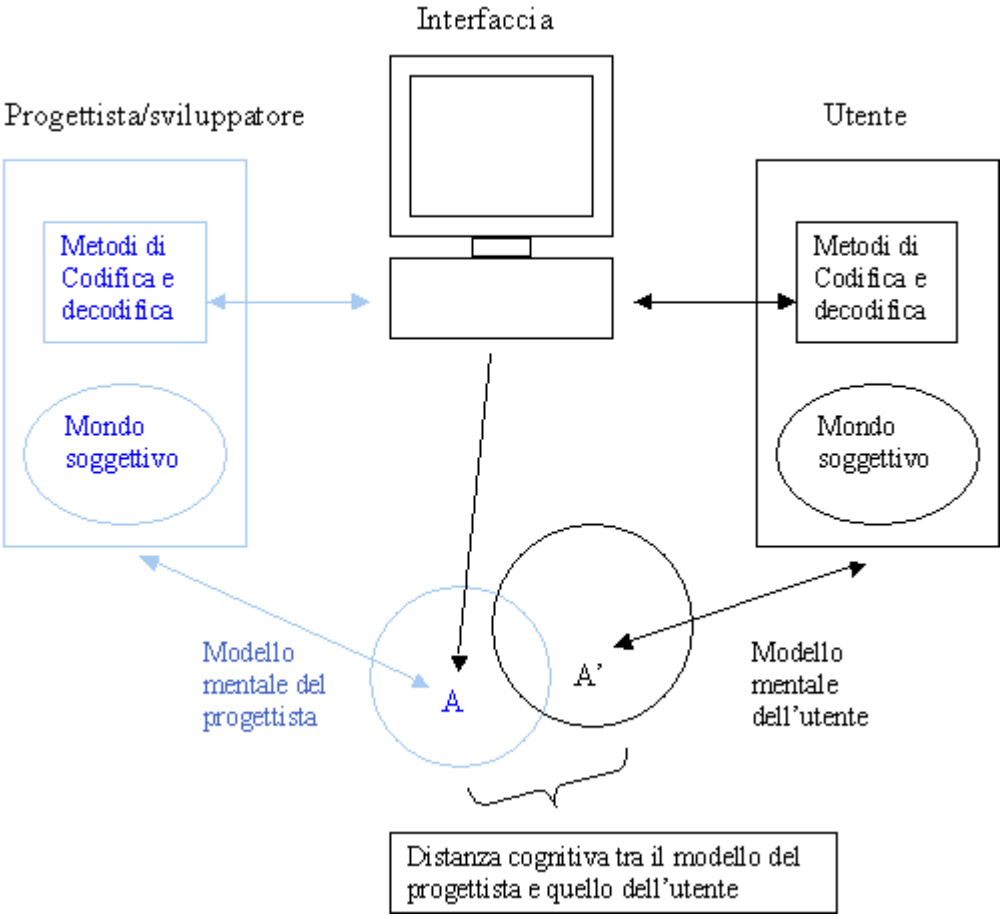
- il modello del progettista, cioè il modello del prodotto e delle sue modalità d'uso che il progettista possiede (in base alle sue conoscenze) e che incorpora nel prodotto, e
- il modello dell'utente, cioè il modello di funzionamento del prodotto che l'utente si costruisce (sempre in base alle sue conoscenze e abilità) e che regola l'interazione con il prodotto stesso.

La comunicazione avviene tra:

- il prodotto software, l'interfaccia, (presente) e il modello del progettista (assente), e
- l'utente (presente).

Comunicazione uomo-interfaccia

Il modello di comunicazione uomo-interfaccia è simile al modello di comunicazione persona-persona. L'interfaccia può essere vista come un messaggio inviato dal progettista-sviluppatore all'utente. Entrambi sono in possesso di un proprio modello mentale, che é una rappresentazione selettiva della realtà e che si evolve con l'esperienza e l'interazione con cose e persone, ed anche posseggono un modello concettuale su ciò che deve fare l'interfaccia. Può essere il modo in cui l'utente concettualizza e capisce l'interfaccia e ciò che il progettista concettualizza e incorpora nell'interfaccia.



Dallo schema precedente si vede che la comunicazione avviene tra utente e interfaccia (che sono sempre presenti). Il progettista ha previsto che certe azioni portino ad risultato definito. Anche l'utente pensa che certe azioni portino ad un certo risultato ma lo scambio di messaggi con l'interfaccia produce i risultati che il progettista ha previsto e non quelli che pensa l'utente.

Infatti, se si indica con **A** il significato di azioni che il progettista ha previsto e incorporato nell'interfaccia (secondo il proprio modello mentale e concettuale) e con **A'** il significato delle stesse azioni dell'utente sull'interfaccia (secondo il proprio modello mentale e concettuale), allora se:

- A e A' coincidono: l'interfaccia risponde in modo adeguato, utente e progettista usano la stessa semantica,
- A e A' non coincidono: le azioni dell'utente non sono quelle previste dal progettista e l'interfaccia non risponde in modo adeguato, utente e progettista usano semantiche diverse.

In quest'ultimo caso è necessario che l'utente inizi un processo di apprendimento sulle reali funzioni e limiti dell'interfaccia.

Obiettivo desiderabile dell'usabilità è quello di rendere la tecnologia sottostante il più possibile invisibile, trasparente all'utente, il quale deve potersi concentrare esclusivamente sul compito, anziché sull'interfaccia.

In sintesi, quindi, per essere usabile, un prodotto software deve:

- essere adeguato ai bisogni e alle aspettative degli specifici utenti finali che lo usano in determinate condizioni;
- risultare facile da capire, da imparare, da usare, ed essere gradevole;
- consentire di eseguire le specifiche attività lavorative in modo corretto, veloce e con soddisfazione.

Compatibilità uomo-computer

L'usabilità dell'interfaccia (prodotto software) non dipende, in generale, dal computer ma dalla compatibilità che si costruisce tra questi due soggetti.

Infatti la compatibilità cognitiva uomo-computer (e quindi uomo-interfaccia) deve rispettare tre regole:

- un'interfaccia deve essere fisicamente compatibile con la morfologia e le caratteristiche della percezione e dell'azione umana; tutto ciò che passa attraverso i nostri sensi più coinvolti (vista, tatto, udito), cioè dalla dimensione dei caratteri, dal colore, dallo scorrimento del testo, dai tempi di risposta ad un comando, oltre alla collocazione spaziale della stazione di lavoro.
- un'interfaccia deve anche essere compatibile con le caratteristiche della comunicazione, della memoria e del modo umano di risolvere i problemi; con l'interfaccia si ha spesso una specie di comunicazione interpersonale, umanizzata, che necessita di interazione e ridondanza, la memoria interessa sia la modalità a breve e a lungo termine, e infine, gli esseri umani risolvono i problemi attraverso tante strategie e quella di prova ed errore è, in questo caso, molto usata.
- un'interfaccia deve agire in preciso contesto; questa puntualizzazione consente di restringere e delimitare l'area di applicazione dell'usabilità.

Le norme ISO

L'usabilità del software è anche regolata dalle norme ISO e in particolare dello standard ISO 9241.

La **parte 10** definisce i principi che caratterizzano il dialogo uomo-computer. Secondo questa norma, il **dialogo** deve essere progettato in modo tale che risulti:

- idoneo al compito;
- autodescrittivo;
- controllabile dall'utente;
- conforme alle aspettative dell'utente;
- tollerante agli errori;
- idoneo alla personalizzazione;
- idoneo all'apprendimento.

La **parte 11** definisce l'usabilità come "il **grado** in cui un prodotto può essere usato da **specifici utenti** per raggiungere **specifici obiettivi** con

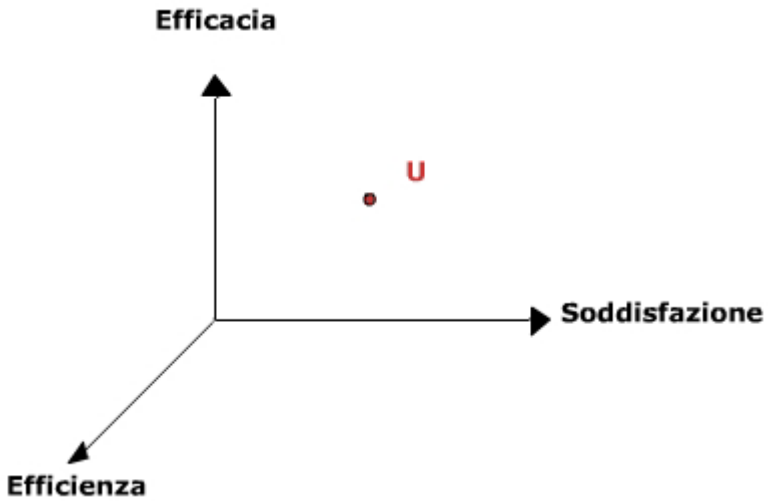
- efficienza, ovvero l'accuratezza e la completezza dei risultati raggiunti;
- efficacia, ovvero la quantità di risorse impiegate per raggiungere l'obiettivo;
- soddisfazione, ovvero il comfort e l'accettabilità del sistema di lavoro da parte degli utenti;

in uno **specifico contesto d'uso**".

Quest'ultima definizione (con la triplice ricorrenza dell'aggettivo "specifico") particolarizza il concetto di usabilità del software. Infatti chi progetta interfacce software, siti web inclusi, deve sempre chiedersi:

- a chi si rivolge, a quale tipologia di utenza
- a cosa serve, per quali scopi è fatto
- dove si usa, in quali situazioni.

Questi tre parametri possono essere in qualche modo misurati (almeno empiricamente) e possono dar luogo ad una misura dell'usabilità, sempre tenendo conto delle particolari specificità dell'utente, del compito e del contesto.



La distanza di U dall'origine del sistema rappresenta, in questo caso teorico, il valore dell'usabilità attribuita all'interfaccia.

I principali standard ISO sull'usabilità

ISO/IEC 9126
Information technology - Software product evaluation - Quality characteristics and guidelines for their use.
ISO 9241
Ergonomic requirements for office work with visual display terminals (VDTs).
ISO 13407
Human-centered design processes for interactive systems.

Nielsen e le sue 10 euristiche

Le 10 euristiche di Nielsen sull'usabilità delle interfacce, derivano dall'applicazione di tecniche di analisi fattoriale su 249 problemi di usabilità.

1 – Visibilità dello stato del sistema
Il sistema deve sempre tenere informato l'utente su cosa sta facendo, fornendo un adeguato feedback in un tempo ragionevole.
Sapere se un oggetto è un link e dove porta
Icona o testo sotto intensificato significa che la funzione non è disponibile
Presenza di un segnale di attività in corso (clessidra, barra di caricamento, messaggio testuale, etc.)

2 – Corrispondenza tra sistema e mondo reale
Il sistema deve parlare il linguaggio dell'utente, con parole, frasi e concetti a lui familiari.
Uso di messaggi testuali, icone, azioni dal significato condiviso da tutti ("salva con nome", icona "cestino", azione "copia e incolla"
Garantire l'associazione tra oggetti e informazione

3 – Controllo e libertà
L'utente deve avere il controllo del contenuto informativo e muoversi liberamente tra i vari argomenti.
Evitare procedure costrittive troppo lunghe (iscrizioni)
Evitare percorsi predefiniti senza possibili scorciatoie
Evitare azioni non volute dall'utente (apertura automatica di pagine non richieste)

4 – Consistenza e standard
L'utente deve aspettarsi che le convenzioni del sistema siano valide per tutta l'interfaccia.
Riportare in ogni pagina alcuni elementi di riconoscimento (logo, stile grafico, etc.)
Dare la sensazione di essere sempre nello stesso ambiente

5 – Prevenzione dell'errore
Evitare di porre l'utente in situazione ambigue, critiche e che possono portare all'errore.
Dare la possibilità di tornare indietro
Evitare che la non comprensione induca in errore

6 – Riconoscimento anziché ricordo
Le istruzioni per l'uso del sistema devono essere ben visibili e facilmente recuperabili.
Produrre layout semplici e schematici
Non contare sulla capacità dell'utente di ricordare il posizionamento degli oggetti che caratterizzano le pagine
Evitare che l'utente riscopra ogni volta l'interfaccia

7 – Flessibilità d'uso
Offrire all'utente la possibilità di un uso differenziale (a seconda della sua esperienza) dell'interfaccia.
Offrire una navigazione gerarchica per i meno esperti
Offrire delle scorciatoie per i più esperti

8 – Design e estetica minimalista
Dare maggior importanza al contenuto che all'estetica.
Evitare di accentuare oggetti irrilevanti o raramente necessari (immagini grandi, etc.)
Evitare che il contenuto informativo della pagina sia messo in secondo piano
Evitare che l'utente si distragga o si confonda

9 – Aiuto all'utente
Aiutare l'utente a riconoscere, diagnosticare e recuperare l'errore.
I messaggi di errore devono essere espressi in linguaggio comprensibile (senza codici)
I messaggi di errore devono indicare in modo preciso il problema e suggerire una soluzione
Chiedere conferma per un'azione importante

10- Documentazione
Anche se il sistema dovrebbe essere usabile senza documentazione è preferibile che essa sia disponibile
Deve essere facile da reperire
Focalizzata sul compito dell'utente
Strutturata in un insieme di passi comprensibili

La valutazione dell'usabilità

Euristica: verifica del rispetto dei requisiti di usabilità da parte di esperti.

Walkthrough: metodo ispettivo. Un gruppo di esperti, progettisti, tecnici, utenti esprimono una valutazione sugli effetti dell'interfaccia sull'utente finale. E' importante che il prodotto risulti facile da apprendere.

Cooperativa: consiste nell'osservare il "pensiero ad alta voce" dell'utente nello svolgimento del compito mediante l'uso dell'interfaccia. Può essere condotto in laboratorio. Esiste interazione tra utente e valutatore.

Test di usabilità: osservazione in ambiente controllato dell'interazione tra utente e interfaccia nell'esecuzione di un compito prefissato e successiva analisi del comportamento. Sono possibili misure quantitative.

Questionari: strumenti di valutazione somministrabili a molti utenti, con tutte le cautele del caso. Implica la conoscenza sommaria dell'interfaccia ed è realizzabile attraverso l'esecuzione, da parte di un campione di utenti, di una serie di compiti, uguali per tutti, e successiva compilazione del questionario.

I requisiti emergenti dell'usabilità HMI

" Una interfaccia HMI è usabile quando soddisfa i bisogni informativi dell'utente finale che la sta utilizzando ed interrogando, fornendogli facilità di accesso e di navigabilità e consentendo un adeguato livello di comprensione dei contenuti. Nel caso non sia disponibile tutta l'informazione, la gestione del processo perde efficienza."

Navigabilità: esistenza di un sistema di navigazione e di orientamento tra le pagine grafiche del sistema. Si deve evitare il senso di smarrimento che spesso è provocato da pulsanti di navigazione non chiari e disomogenei. L'utente deve sapere dove si trova e come può ritornare facilmente ad un punto precedente.

Utilità attesa: disponibilità di informazioni e/o servizi che corrispondono alle aspettative degli utenti. L'utente ha delle aspettative di ritorno per il tempo che dedica alla visita del sito. Bisogna evitare che le promesse del sito siano disattese o addirittura false.

Completezza dei contenuti: presenza di contenuti informativi a livello di dettaglio necessario per gli utenti. E' molto difficile che un sito soddisfi il bisogno informativo di ogni tipologia di utenti in una sola pagina di contenuto. Occorre guidare l'utente con informazioni generali e link ad informazioni di dettaglio, visibili solo quando interessa specificamente. E' importante che l'ampiezza di contenuti e il loro livello di dettaglio si adattino ad ogni tipologia e siano raggruppati in modo chiaro e completo. A volte i contenuti vanno raggruppati tra loro in modo diverso nello stesso progetto, in funzione specialistica definita dall'audience a cui si rivolge.

Comprensibilità delle informazioni: la forma e la qualità con cui l'informazione e i contenuti vengono presentati nelle pagine grafiche del sistema. Molto importante il linguaggio usato, soprattutto per operazioni interattive. Deve esistere un sistema di classificazione delle informazioni comprensibile da tutti, anche se il contenuto finale può essere specialistico.

Efficacia comunicativa: la strategia comunicativa del progetto. L'efficacia comunicativa è una misura della credibilità del sistema di interfaccia e si basa sia sul marchio dell'istituzione/struttura che rappresenta, sia sulla capacità di essere chiari ed esaustivi per portare ad una relazione di fiducia con gli utenti.

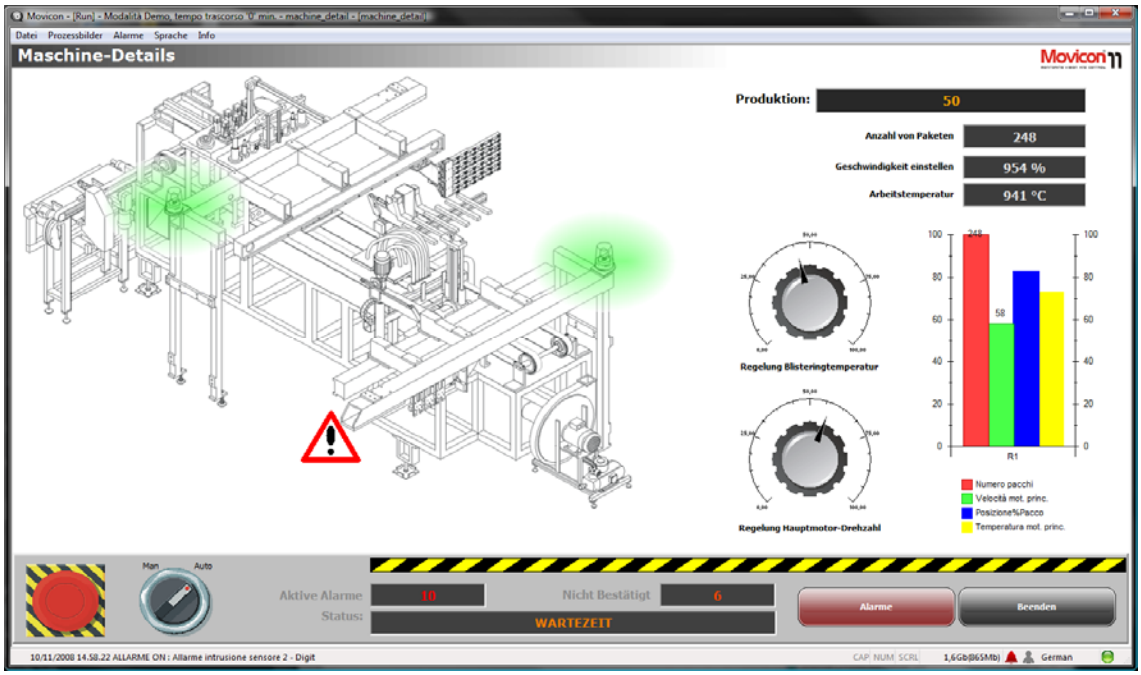
Attrattività grafica: la qualità della grafica e la piacevolezza visiva del sito. La grafica deve portare ad un giusto equilibrio tra emozioni e comfort che induce sull'utente e utilizzo consapevole dei contenuti. Non deve nascondere il vero scopo del sistema.

Velocità dei dati: Il sistema deve aggiornare i dati in "tempo reale". Con questo termine si intende un tempo rapido, adeguato al processo gestito dal sistema. I tempi possono essere considerati accettabili per l'utente tra i 0 ed i 3 secondi. Questi tempi valgono anche per l'accesso alle pagine richieste. Tempi superiori sono pure accettabili, ma comportano spesso difficoltà per l'utente nel gestire la reattività e diminuiscono il comfort dell'utente nel gestire il processo.

I principi di usabilità delle interfacce software

Fondati sul modo di ragionare e operare delle persone quando interagiscono con un prodotto software, i principi di usabilità rappresentano un riferimento importante di cui tenere assolutamente conto sia in fase di progettazione che in fase di valutazione dell'usabilità (Jakob Nielsen ha basato sui cosiddetti 'principi euristici' un metodo di valutazione noto, appunto, come valutazione euristica).
I principi riportati in queste pagine possono essere considerati una discreta sintesi di quanto si può rintracciare in letteratura su questo argomento:

- realizzare un dialogo semplice e naturale
- semplificare la struttura dei compiti
- agevolare il riconoscimento piuttosto che il ricordo
- fornire feedback in modo da rendere visibile lo stato del sistema
- prevenire gli errori di interazione e facilitarne il recupero
- essere consistenti
- parlare il linguaggio dell'utente
- agevolare la flessibilità di utilizzo e l'efficienza dell'utente
- fornire help e manuali



Realizzare un dialogo semplice e naturale

Il prodotto software deve proporre all'utente un modello di dialogo che sia coerente con il modello dell'attività dell'utente e con il suo sistema di attese e che, in ogni caso, sia facilmente intuibile interagendo con il prodotto stesso.

La progettazione di un dialogo semplice e naturale è agevolata principalmente da due fattori:

- dalla continua considerazione del modo di lavorare dell'utente, dalle sue caratteristiche e dalle sue esigenze;

- dalla scelta di soluzioni di interfaccia che tengano conto soprattutto dei principi sui compiti, sul riconoscimento, sul feedback e sugli errori.

Alcuni modi per rispettare il principio del dialogo semplice e naturale sono:

- tener conto del modo di operare dell'utente e del suo modello dell'attività nella:
  - organizzazione dei contenuti e della struttura del sistema,
  - implementazione della logica dell'interazione,
  - definizione dell'ordine dei menu,
  - definizione dell'ordine di presentazione delle finestre o delle pagine,
  - organizzazione dei contenuti e degli oggetti all'interno delle finestre o delle pagine;
- fornire un buon modello concettuale del sistema e renderne evidenti l'organizzazione, la logica d'interazione, etc... affinché l'utente possa prevedere gli effetti delle proprie azioni: interagire con un prodotto senza avere compreso e sviluppato un modello di come funziona è come agire alla cieca, senza sapere cosa succederà e cosa aspettarsi a ogni azione;
- rendere evidenti le informazioni rilevanti di cui l'utente ha effettivamente bisogno per svolgere l'attività, evitando di presentare informazioni superflue o che si usano raramente: le informazioni irrilevanti possono entrare in 'competizione' quelle fondamentali e togliere a queste ultime la necessaria visibilità;
- anticipare quanto possibile le informazioni che l'utente troverà nella pagina successiva o quello che succederà a seguito di una azione.

Semplificare la struttura dei compiti

I compiti o le attività che l'utente deve svolgere in interazione con il sistema devono avere una struttura semplice. Devono essere progettati e implementati nel sistema riducendo al minimo la necessità di elaborazione delle informazioni da parte dell'uomo dovute all'utilizzo dello strumento informatico.

Donald Norman suggerisce quattro approcci per semplificare i compiti:

- mantenere il compito invariato, ma offrire sussidi mentali;
- usare la tecnologia per rendere visibile quello che altrimenti sarebbe invisibile;
- automatizzare, mantenendo il compito sostanzialmente inviariato;
- cambiare la natura del compito.

I primi tre approcci prevedono di non cambiare sostanzialmente il compito che l'utente deve fare. Nel primo e nel secondo caso, il suggerimento è quello di offrire sussidi esterni (che supportano le capacità cognitive dell'utente non costringendolo a dover ricordare tutto a mente) e feedback (che consente all'utente di controllare le componenti non visibili del sistema al fine di verificarne l'adeguato funzionamento). Ad esempio, nella scrittura a videoterminale, la struttura del compito è sostanzialmente identica alla scrittura manuale. In più, però, la scrittura a videoterminale, grazie al sussidio del correttore automatico che segnala gli errori, consente di migliorare la qualità del lavoro. Nel terzo caso, pur mantenendo invariata la struttura del compiti, alcune sue parti (quelle più rischiose o complicate) vengono eseguire dalla tecnologia e non più dall'utente.
L'ultimo approccio, infine, quando il compito è intrinsecamente complesso per le abilità richieste, prevede una riprogettazione sostanziale del compito: l'obiettivo rimane, ovviamente, lo stesso, ma il modo in cui tale obiettivo viene raggiunto è totalmente diverso.

Agevolare il riconoscimento piuttosto che il ricordo

Osservando l'interfaccia, l'utente deve poter capire cosa deve fare, come può farlo e, una volta eseguita una azione, deve poter capire cosa è successo e quali sono stati i risultati.

Dal momento che è più facile riconoscere e ricordare una cosa vedendola direttamente, piuttosto che recuperare l'informazione dalla memoria, il modo più semplice per agevolare l'utente è quello di rendergli visibili le cose sull'interfaccia, ovvero fornirgli dei sussidi esterni che gli agevolano il ricordo.

Dire che le cose devono essere visibili sembra banale, ma non lo è! Ecco alcuni esempi di disattenzione di questo principio:

- essere costretti ad imparare a memoria certi comandi o informazioni (o ricorrere alla guida operativa o chiedere suggerimenti a colleghi e amici più esperti);
- non capire, osservandolo, se un elemento (un campo di edit, un link, un pulsante, etc...) è selezionabile/modificabile oppure no;
- non capire se l'azione richiesta è stata eseguita dal sistema;
- non capire perché l'azione richiesta non è stata eseguita dal sistema;
- navigando nelle pagine, non capire più dove ci si trova e da dove si era partiti.

Alcuni suggerimenti per agevolare il riconoscimento:

- sfruttare il 'mapping' naturale, ovvero la correlazione naturale che esiste tra due cose, tra causa ed effetto, tra comandi, loro azionamento e risultati (un esempio di 'mapping' naturale è la manipolazione diretta degli oggetti. Con queste tecniche, l'utente non è costretto a ricordare il modo di utilizzare degli oggetti o a descrivere le azioni da eseguire: semplicemente le esegue direttamente sullo schermo, spostando, ad esempio, un documento dalla scrivania virtuale del suo PC al cestino, così come farebbe nel mondo reale);
- fare in modo che le azioni consentite sull'interfaccia siano chiaramente visibili;
- rendere evidente lo stato del sistema ad ogni momento e ad ogni azione dell'utente (risultati dell'azione svolta, contesto nel quale ci si trova, etc...);
- dotare tutte le pagine di un titolo significativo che illustri adeguatamente il tipo di informazione visualizzata o le azioni da svolgere sulla finestra o sulla pagina;
- utilizzare un linguaggio e una grafica corretta e significativa per l'utente, in modo da non costringerlo a interpretazioni del significato;
- usare liste di selezione che ricordano le scelte ammissibili e il formato consentito;
- fornire delle informazioni di anteprima sugli oggetti selezionati;
- dotare le icone e i simboli grafici di tool tip descrittive della funzionalità associata all'icona stessa;
- abilitare o disabilitare i comandi in base al contesto operativo, in modo da ricordare all'utente l'obbligatorietà di certe azioni o la relazione sequenziale tra certe altre;
- rispettare la consistenza nell'organizzazione dei contenuti e degli oggetti all'interno delle finestre e delle pagine, in modo da non costringere l'utente a continue osservazioni dell'intera finestra o pagina per ritrovare determinati oggetti o gruppi di informazioni.

Fornire feedback in modo da rendere visibile lo stato del sistema

Il feedback rappresenta l'informazione di ritorno in risposta all'azione che l'utente ha eseguito sulla interfaccia e ha lo scopo di rendere visibile all'utente lo stato corrente del sistema, in modo da evitare errori, incomprensioni e blocchi durante l'interazione.
Inteso in questo modo, il feedback non si riferisce solo alle azioni sbagliate da parte dell'utente e alla relativa messaggistica di errore, ma coinvolge tutti i modi per comunicare all'utente cosa sta facendo il sistema al momento corrente:

- quale azione ha compiuto o sta compiendo l'utente;
- quali sono gli effetti della propria azione sul prodotto;
- il nuovo stato del prodotto a seguito dell'azione effettuata.

Oltre alla messaggistica, prevalentemente usata per correggere errori di interazione, i modi per fornire feedback sullo stato corrente del prodotto sono estremamente svariati:

- osservare sulla finestra lo scorrimento dell'oggetto selezionato quando l'utente effettua una azione di 'drag & drop' informa che l'esecuzione dell'azione sta avvenendo correttamente;
- il cambio di forma del puntatore del mouse a seguito della selezione di qualche strumento grafico informa che sono possibili solo certe operazioni e non è possibile, ad esempio, digitare un testo;
- la comparsa di un indicatore di avanzamento informa che per eseguire una determinata operazione da parte del prodotto è necessario un certo tempo;
- l'enfasi di non disponibilità di un oggetto indica che quello strumento non è disponibile o quell'azione non è consentita;
- nessun effetto sull'interfaccia abbinato all'emissione di un segnale sonoro indica che si sta cercando di compiere una azione non consentita allo stato corrente. E così via.

Ma entro quanto tempo deve essere fornito il feedback? Le nostre considerazioni per i sistemi industriali, sulla base di quanto verificato e studiato da Jakob Nielsen:

- **0,3 secondi** è, approssimativamente, il tempo per dare all'utente la sensazione che il sistema ha reagito istantaneamente;
- **2 secondi** è, approssimativamente, il tempo massimo per mostrare i risultati dell'azione eseguita dall'utente, anche se noterà un ritardo nella riposta del sistema;
- **5 secondi** è, approssimativamente, il tempo massimo per eseguire accessi alle pagine meno frequentate del sistema. Le pagine più frequentate devono rientrare nei tempi di cui al punto sopra.
- **7 secondi** è, approssimativamente, il tempo massimo per accedere a dati storici di routine, per eseguire analisi fuori linea su filtri e database in archivio (esempio storico eventi o trends). L'utente noterà il ritardo, ma l'azione non incide sul processo gestito.
- **10-20 secondi** è, approssimativamente, il tempo massimo per accedere a dati storici consistenti, per eseguire analisi fuori linea su filtri e database in archivio (esempio estrazione dati per report, statistiche o altro). L'utente noterà il ritardo, ma consapevolmente in quanto sta facendo azioni che non incidono sul processo gestito.
  In caso di archivi molto consistenti, è opportuno informare l'operatore che l'operazione è in corso e richiede tempo, consentendogli di annullare tale operazione se eventualmente richiesto dal processo.

I tempi di risposta possono dipendere da molte cose (performance dei server e memoria disponibile, tipo e velocità di connessione al PLC, quantità di informazione che deve essere trasferita, etc...), ma gli utenti non sono interessati a queste motivazioni. Se l'attesa è lunga, gli utenti pensano solo che non viene offerto loro un buon servizio e il livello di fiducia nel fornitore è destinato a diminuire.

Prevenire gli errori di interazione e facilitarne il recupero

Commettere errori nell'interazione con un prodotto è naturale. Anzi, qualsiasi errore che possa teoricamente essere commesso, prima o poi accadrà! Ogni azione dell'utente va concepita come un tentativo verso una giusta direzione. L'errore non è altro che una azione specificata in modo incompleto o inesatto. Si tratta di una componente naturale del dialogo utente-sistema che va tollerata, garantendo la giusta flessibilità di utilizzo che consente agli utenti di navigare liberamente senza entrare in vicoli ciechi e in situazioni critiche. Ci sono alcuni tipi di errori che sono difficilmente eliminabili, come le sviste: inconsapevolmente, viene eseguita un'azione diversa rispetto a quella che ci proponeva nelle intenzioni, a causa di una distrazione o di una interruzione.

Altri tipi di errore, invece, si possono prevenire con una buona progettazione dell'interfaccia: sono gli errori commessi a seguito di una applicazione sbagliata di regole di interazione o per la mancanza di sufficienti e adeguate informazioni e conoscenze. Rientrano tra questi tipi di errore quelli dovuti ad un modello di dialogo che l'utente non capisce o che non incontra le sue aspettative e per questo egli applica regole di interazione sbagliate rispetto a quelle richieste dal prodotto.
Il contributo fondamentale alla prevenzione degli errori d'interazione deriva, quindi, dal rispetto dei principi sul dialogo, sui compiti, sul riconoscimento e sul feedback sulla cui base è consentito agli utenti di individuare, riconoscere e adeguare le proprie azioni alle possibilità offerte attraverso l'interfaccia. Altri modi di prevenire gli errori prevedono l'utilizzo di funzioni bloccanti, che impediscono la continuazione di azioni sbagliate o che possono portare a risultati distruttivi.
Tuttavia, poiché gli errori sono sempre possibili, è importante anche che il sistema sia progettato in modo da diagnosticarli quando occorrono e facilitarne la correzione.
I modi più semplici per raggiungere questo obiettivo sono:

- fornire funzionalità di annullamento delle operazioni, di ripristino delle condizioni di default o comunque prevedere la richiesta di conferma di operazioni pericolose;
- fornire una messaggistica efficace (vedi anche feedback);
- evitare di presentare pagine senza opzioni di navigazioni;
- rendere sempre disponibili le funzioni per il ripristino programma o per ritornare alla home page;
- fornire comandi per interrompere operazioni molto lunghe (ove possibile).

Essere consistenti

La consistenza si riferisce al fatto che la sintassi (linguaggio, campi di input, colori, etc..) e la semantica (comportamenti associati agli oggetti) del dialogo devono essere uniformi e coerenti all'interno di tutto il prodotto software.
La consistenza permette all'utente di trasferire agevolmente la conoscenza da una applicazione all'altra, aumenta la predicibilità delle azioni e dei comportamenti del sistema e ne favorisce l'apprendibilità.
Un problema di consistenza è relativo ai font ed ai menu o link di navigazione.
Molto spesso, all'interno dello stesso programma, si vedono pagine con font diversi per dimensione, stile e colori. Analogamente, i link vengono proposti in svariati formati e colori, senza limiti alla fantasia.
Le pagine grafiche sono da immaginare come pagine di un libro. E non si è mai visto un libro 'importante' che usa caratteri diversi per ogni paragrafo, titoli di dimensioni e colori differenti posizionati una volta centrali e una volta a sinistra sulla testata della pagina, rimandi ad approfondimenti ogni volta presentati con uno stile diverso!
L'inconsistenza nei font, nella struttura dalla pagina, nella grafica genera una situazione di confusione, fornisce l'impressione di una mancanza di cura e di attenzione e, in definitiva, di professionalità.
In sintesi, la consistenza deve essere garantita a diversi livelli:

- **consistenza del linguaggio e nella grafica**: la stessa parola, la stessa icona, lo stesso colore devono identificare lo stesso tipo di informazione o lo stesso tipo di azione entro tutto il prodotto;
- **consistenza degli effetti**: gli stessi comandi, le stesse azioni, gli stessi oggetti devono avere lo stesso comportamento e produrre gli stessi effetti in situazioni equivalenti; non associare agli stessi comandi, azioni e oggetti comportamenti diversi;
- **consistenza nella presentazione**: gli stessi oggetti o lo stesso tipo di informazioni devono essere collocati tendenzialmente nella stessa posizione, avere la stessa forma e lo stesso ordine;
- **consistenza tra ambienti applicativi**: una applicazione, un sito non sono mondi isolati! Gli utenti utilizzano differenti applicazioni e navigano tra siti diversi, e imparano come

funzionano certi oggetti di interfaccia. Entrando nella nostra applicazione/sito si aspettano di ritrovare la stessa tipologia di oggetti che si comporta nel modo che hanno imparato. Soluzioni alternative, utilizzo non convenzionale degli oggetti grafici inducono solo incertezza di utilizzo e aprono la porta agli errori di interazione.

Parlare il linguaggio dell'utente

Il linguaggio utilizzato a livello di interfaccia deve essere semplice e familiare per l'utente e rispecchiare i concetti e la terminologia a lui noti.
Va evitato il più possibile un linguaggio tecnico e orientato al sistema che utenti non esperti di informatica possono non comprendere e termini stranieri. Vanno evitate, ad esempio, parole come 'default', 'directory', o frasi del tipo 'il documento è trasferibile via ftp'.
Rientrano tra gli elementi del linguaggio anche le icone e le metafore, forme per rappresentare concetti in forma grafica e simbolica, che, se ben realizzate, possono agevolare la comprensione in modo più efficace e diretto rispetto alle parole.
La definizione di un linguaggio adeguato e significativo per l'utente, soprattutto se simbolico, è comunque un lavoro ben più difficile di quanto generalmente si pensi poiché comporta conoscere molto bene il mondo degli utenti.

Due consigli:

- verificare con gli utenti la comprensibilità del linguaggio (etichette, istruzioni, elenchi, etc...);
- usare preferibilmente icone e metafore già sperimentate e consolidate; se si vogliono usare simboli originali è sempre meglio sottoporli prima a test di comprensibilità con gli utenti finali del prodotto.

Agevolare la flessibilità d'utilizzo e l'efficienza dell'utente

Pensare alla produttività dell'utente! Nella definizione degli strumenti in grado di agevolare la flessibilità e l'efficienza, va considerato che le esigenze degli utenti variano in relazione al loro livello di esperienza rispetto al compito e alle tecnologie informatiche. Ne consegue che, in relazione a questi due aspetti, il livello di supporto richiesto, gli strumenti utilizzati e le strategie di interazione messe in atto dagli utenti possono essere diverse. Gli utenti non esperti, ad esempio, amano essere guidati passo per passo, mentre gli utenti più esperti preferiscono utilizzare scorciatoie, delle quali anche utenti non esperti man mano che aumenta il loro livello di esperienza possono usufruire.

Si può agevolare la flessibilità e l'efficienza d'uso fornendo, ad esempio:

- 'facilities' d'inserimento (ad esempio, l'anticipazione da parte del sistema nell'inserimento di un termine) e acceleratori (una combinazione di tasti, come ad esempio, CTRL V);
- salti nella navigazione che evitano di passare in punti intermedi;
- funzioni di personalizzazione dell'interfaccia, ovvero la possibilità di modificare alcuni aspetti del sistema in base alle esigenze del compito, alle caratteristiche dell'utente e sue preferenze personali. Da notare che, una volta personalizzato, il sistema deve mantenere, alle successive riaperture, le impostazioni date dall'utente. Alcuni esempi di aspetti dell'interfaccia che dovrebbero essere personalizzabili sono:
  - la lingua,
  - la dimensione dei caratteri,
  - le impostazioni di default,
  - il formato dei dati presentati e livello di dettaglio,
  - la disposizione di alcuni oggetti grafici.

Un altro aspetto dell'efficienza è anche il tempo di risposta del sistema alle azioni dell'utente, problema che nelle applicazioni web è tra i più critici.
I tempi di risposta ad ogni chiamata dell'utente possono dipendere da molte cose (performance dei server e della rete, tipo e velocità di connessione, quantità di informazione che deve essere trasferita, etc...), ma gli utenti non sono interessati a queste motivazioni. Se l'attesa è lunga, gli utenti pensano solo che non viene offerto loro un buon servizio e il livello di fiducia nel fornitore è destinato a diminuire.

Fornire help e manuali

L'argomento della documentazione (help in linea o manuali utente) è piuttosto controverso per diversi motivi:

- un buon prodotto, teoricamente, non dovrebbe richiedere la consultazione della documentazione;
- la documentazione viene spesso usata per compensare eventuali problemi di usabilità del prodotto;
- nella maggior parte dei casi, gli utenti ignorano questi strumenti di supporto.

Per quanto riguarda l'ultimo punto, infatti, generalmente gli utenti ricorrono all'help in linea o alla documentazione solo come ultimo tentativo, cercando (e non trovando quasi mai) la soluzione al proprio caso specifico. Nella lettura delle informazioni riportate, inoltre, tendono a non approfondire gli argomenti, leggendo rapidamente solo poche righe. Infine, come per tutti i testi scritti, la comprensibilità di help e manuali, se non accuratamente verificata con gli utenti finali, non è sempre garantita.

Considerati questi aspetti, quando la documentazione può essere necessaria, essa va realizzata con l'obiettivo di garantire:

- facilità di consultazione,
- comprensibilità e brevità dei testi,
- orientamento all'attività dell'utente,
- efficacia nella risoluzione del problema.

Documento Redatto da Progea Srl
Modena, 15 Giugno 2008

Fonti:

- *Progea Srl*
- *Dott.sa Cinzia Stortone,*
- *FAR - Università Torino*
