

# Interazione Uomo-Macchina

879276 \* A.A. 2023/2024



# Interazione Uomo-Macchina

Appunti del Corso ✎ Anno Accademico 2023-2024 ✎ 879276

---

**26.09.2023**

La disciplina di Interazione Uomo Macchina si occupa di studiare come progettare e valutare interazioni tra utenti e strumenti informatici, analizzando come influenzare tale interazione in vari modi

Il team principale di questo corso è l'**utente** in varie declinazioni, specificatamente l'essere umano, la persona, con i suoi bisogni e preferenze

Un obiettivo del corso è imparare l'abilità di valutare l'**usabilità** a livello professionale

Due elementi distintivi di questo corso in particolare rispetto a quelli di altre Università sono lo studio della **semiotica** e del **persuasive computing**

Un'**interazione** è un processo in cui due oggetti agiscono in modo di influenzarsi reciprocamente

**03.10.2023**

## Concetti Base

L'interazione uomo-macchina è una disciplina molto ampia, ci concentreremo sugli aspetti di **usabilità** dei sistemi informatici interattivi

Cosa vuol dire **Human-Computer Interaction?**

HCI (o IUM) è una disciplina che si occupa della **progettazione, realizzazione e valutazione** di sistemi interattivi con capacità computazionali destinati all'**uso umano**; e delle studio dei principali fenomeni che li circondano

Si tratta dunque di sistemi computazionali reali che verranno utilizzati da persone reali per svolgere azioni reali.

L'IUUM è anche IUUUM ovvero, **Interazione Uomo-Uomo Mediata dalla Macchina**, ci occuperemo dunque anche del contesto di tutti quei sistemi di rete che connettono le persone.

Con l'introduzione di linguaggi di alto livello come Fortran o COBOL negli anni 60, i primi elaboratori IBM cominciando a prendere piede, inizialmente in ambienti di lavoro, con un numero limitato di funzioni e interfacce, successivamente anche in ambienti domestiche con tanto di monitor e tastiera, e nel 1984 Apple annuncia il Macintosh, il primo elaboratore con un'interfaccia interamente visuale.

La disciplina nasce dunque negli anni 80, la prima conferenza di CHI (Computer Human Interaction) avviene nel 1983.

Nasce dall'intersezione di due differenti discipline: **informatica** e **ergonomia**, attingendo da vari domini scientifici:

### **Scienze dei computer, scienze della progettazione, scienze dell'uomo**

Si può differenziare tra **ergonomia**, studio delle attività, e **ergonomia cognitiva**, studio dell'interazione tra l'uomo e strumenti per elaborazione studiando i processi cognitivi (**linguaggio** e **emozioni**)

Progettare per l'interazione è quindi progettare per l'**utente**, per utenti che fanno qualcosa in un determinato ambiente e con determinati effetti sul mondo.

Si individuano **tre dimensioni** per cui è difficile progettare per l'essere umano

1. La varietà dei **sistemi interattivi**
2. La varietà degli **utenti**
3. La varietà degli **scopi e degli usi**

Dobbiamo quindi studiare come fare a progettare per la **varietà**, siamo interessati a quello che si trova al di fuori del workflow

Possiamo riassumere i temi principali trattati nell'HCI con:

- Criteri per la **progettazione dell'interazione** fra esseri umani e sistemi interattivi
- Criteri per la **valutazione della usabilità** dei sistemi interattivi
- Progettazione di nuove **tecniche di interazione**
- Valutazione dell'**impatto** (breve, nel "qui e ora", e nel lungo periodo, sulla collettività → conseguenze inattese) dell'automazione nei contesti umani
- Valutazione di un **sistema socio-tecnico**

Gli standard tecnici riguardo l'usabilità sono specificati nell'ISO-9241

**Non esistono metodologie per insegnare a progettare una "buona" interazione!**

È importante notare che l'usabilità è un tipo di effetto sugli utenti, non una caratteristica intrinseca del sistema, ma una **caratteristica dinamica**. In questa disciplina è quindi necessario il coinvolgimento nella vita reale degli utenti, non tutti in quanto è impossibile, ma "un po'" di utenti statisticamente rappresentativi di "tutti"

Un **sistema socio-tecnico** è per prima cosa un sistema, ovvero un insieme di componenti ben organizzati l'uno in relazione con l'altro. È un insieme di elementi irrelati ed eventualmente mutualmente dipendenti che, agli occhi di un osservatore esterno, appaiono come una entità unitaria ma collettiva

È un sistema in cui la componente umana (**sociale**) e quella tecnica (**tecnologica**) sono inestricabilmente legate tra di loro, e la loro **interazione** porta a fenomeni **emergenti impredicibili**

Un comportamento viene detto emergente quando la sua proprietà non è direttamente derivabile dalle sue parti

Le proprietà **funzionali** riguardano il **funzionamento** dell'intero sistema una volta che tutte le sue parti, assemblate come devono, funzionano bene

Le proprietà che si dicono invece **non funzionali** riguardano **quanto bene** opera il sistema in un determinato **ambiente o contesto** (comfort, reliability, usability, safety, security, performance...)

10.10.2023

**STS thinking** è un approccio consapevole che le due componenti si integrano e ottimizzano solo in congiuntione **subottime (joint optimization)**

Job satisfaction

Workers' needs

Skill enchantment

L'introduzione di una tecnologia in un contesto sociale è **parte di un cambiamento più ampio**. Non c'è dicotomia tra persone e tecnologie, ma elementi dell'uno sono parte dell'altro e viceversa

Un sistema socio-tecnico è **complesso**, in cui è impossibile prevedere ogni evoluzione  
Si parla di **conseguenza inattesa** delle tecnologia

È difficile prevedere l'outcome di azioni, spesso l'introduzione di una misura per raggiungere un certo obiettivo possono produrre **backfire** (effetto Peltzman)

Quando si progetta qualcosa che è **trop**po buono, si verifica una situazione di **OverReliance**, di cui si possono distinguere due sotto-categorie

OverReliance		
OverDependence	OverConfidence	
Mancanza di autonomia	Pensare che non andrà mai giù	
Abuso al di là dei bisogni	Pensare che non ne deriverà mai un danno	
Mancanza, dimenticanza e ignoranza	Pensare che non può sbagliare mai	

La **complacency** e l'**automation bias** sono altri due concetti simili, il primo è una eccessiva fiducia per cui un sistema computazionale funzionerà sempre per come è stato progettato, mentre il secondo è una eccessiva fiducia nella risposta dei sistemi di supporto decisionale

### ~ Recap ~

Progettare sistemi usabili è progettare per l'uso, che significa progettare per ambienti che il progettista non controlla e non può essere previsto nel dettaglio (sistemi socio-tecnico), l'utente ne è la variabile

Per progettare sistemi usabili non esistono metodologia sistematiche e strutturate, si impara per imitazione di buoni esempi ed esperienza, bisogna essere creativi (effort, skill e capacità di giudizio) e valutare tramite feedback reali il sistema progettato, il tutto tenendo a mente il concetto di conseguenza

Essere progettisti responsabili non significa nient'altro che il proprio sistema sarà il componente di un sistema socio-tecnico più ampio, che può stravolgere o modificare tale sistema

12.10.2023

## (Altri) Concetti Base

### Porte di Norman

Le porte di Norman sono porte progettate così male che non capisci se per aprirle devi spingere o tirare

Questo introduce i concetti di **Discoverability** e **Feedback**

Indica la tendenza del progettista a concepire oggetti che non invitano all'uso (corretto) sulla base di elementi visuali ed indicazioni chiare (**affordances**), basandosi sull'intuito

### Toilet di Floyd

Da 2001 Space Odyssey, la Zero Gravity Toilet era un sistema per il quale era necessario leggerne il manuale prima di usarla

Indica la tendenza del progettista a non semplificare le cose, giustificato dalla presenza di un manuale

## Terminologia dell'IUM

Un **sistema interattivo** è la combinazione di componenti hardware e software che ricevono input da un utente umano, e gli forniscono un output allo scopo di supportare l'effettuazione di un compito

Un'**interfaccia** è l'insieme dei componenti di un sistema interattivo che forniscono all'utente informazioni e comandi per permettergli di effettuare specifici compiti attraverso il sistema

Ciò che accomuna le due definizioni è l'esecuzione di un **compito (task)**

Per l'utente, l'interfaccia è il sistema interattivo stesso

## Constraints

Un vincolo può essere:

**Strutturale** (o passivo) (Form con solo tre campi perché non ne prevedo un quarto)

**Funzionale** (o attivo) (Controlli se l'utente non inserisce tutti i dati necessari)

Sono strumenti utili per garantire migliore qualità dei dati e standardizzazione, ma se usati troppo, o male, si verifica il fenomeno dei **workaround**



Un **workaround** è una qualsiasi azione relativa all'esecuzione di un processo o compito non prevista dal progettista che può bypassare o piegare ai propri fini l'uso del sistema

## Affordance

L'affordance è un concetto contrario, o duale, al constraint; in questo caso, "to afford" prende il significato di offer, yield, provide...

Un'affordance è una qualità o proprietà di un oggetto che definisce i suoi possibili usi o rende chiaro come può o dovrebbe essere usato; qualsiasi proprietà di un oggetto che invita una persona competente all'azione mediata dall'oggetto

Nella progettazione delle interazioni ci sono diversi tipi di affordance:

Affordance **cognitive**, aiutano gli utenti nelle loro azioni cognitive (pensare, decidere, imparare, ricordare...)

Affordance **fisiche**, aiutano gli utenti nelle loro azioni fisiche (cliccare, toccare...)

Affordance **sensoriali**, aiutano gli utenti nelle loro azioni fisiche (vedere, sentire, gustare...)

Affordance **funzionali**, aiutano gli utenti a fare qualcosa, lavorare

Affordance **emozionali** (emoji) e affordance **sociali** (il "like")

Esiste anche il concetto di **user-created affordance**, ovvero "modifiche" effettuate dagli utenti per rendere più chiaro o funzionale l'utilizzo di artefatti

## Mapping

È la relazione tra, da una parte un controllo e l'azione che esso "afforda" e, dall'altra, l'effetto che tale azione avrà nel mondo reale/applicativo

Si parla di mapping **diretto** quando c'è una relazione fisica, spesso legata alla posizione (topologia), e l'effetto, mentre per mapping **arbitrario** è una relazione del tutto immaginata dal progettista e che l'utente addotta solo attraverso esperienza d'uso

Il mapping è una certa forma di affordance; un'affordance non solo suggerisce un'azione sull'artefatto, ma anche l'effetto che avrà, in funzione del suo mapping

"Se l'uso corretto di un certa soluzione di design dipende dalle etichette, potrebbe essere difettoso. Le etichette sono importanti e spesso necessarie, ma l'uso appropriato di mapping naturali può minimizzare il bisogno di etichette. Ogni volta che un'etichetta sembra necessaria, pensate a un modo diverso di progettare la cosa"

Progettare attraverso le affordance significa:

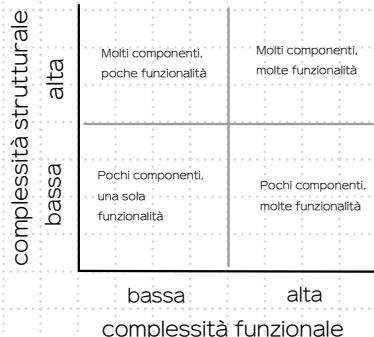
1. Seguire le convenzioni già consolidate o usare mapping naturali o diretti
2. Se appropriato e necessario, usare parole oltre a icone e elementi grafici
3. Usare metafore riconoscibili e chiare
4. Essere consistenti e coerenti con l'uso di un modello concettuale (ovvero i tre punti precedenti)

**13.10.2023**

Una buona interfaccia da benefici in termini di più alta produttività, poi basso turnover, minore frustrazione, più soddisfazione... tutto ciò si traduce in più bassi **costi operativi**.

Con una buona interfaccia, l'utente può svolgere il suo compito dimenticandosi di star usando un computer (teoria del flusso)

## Dimensioni della complessità



Una terza dimensione della complessità, è la **complessità d'uso**

Lo spazio della complessità è quindi tridimensionale, e ci si può immaginare un artefatto come dentro questo spazio

È necessario semplificare l'uso perché, data la pervasività della tecnologia nel mondo di oggi, c'è la necessità di renderla accessibile a tutti, ed è anche necessario comprendere ruoli e possibilità della tecnologia, al fine di migliorare la qualità della vita

Si può intendere l'interfaccia come un **filtro semplificatore** per l'utilizzo del sistema, rispetto alla sua complessità funzionale e strutturale

### ~ Recap ~

Esistono concetti cardine della disciplina che rimangono costanti e non dipendono dalla tecnologia, ovvero: constraint, mapping, affordance e workaround

Ripetiamo che non esistono metodologie per progettare una buona interazione; l'osservazione dei design degli altri e il coinvolgimento degli utenti è la ricetta per imparare

L'osservazione è necessaria per apprendere cosa c'è di buono e cosa di cattivo nei design esistenti, migliorando la propria sensibilità (porte di Norman, toilet di Floyd)

Pensare alle conseguenze (anche inattese) che il mio progetto avrà sugli utenti è necessario! Progettazione responsabile

## Valutazione di Usabilità

La valutazione dell'usabilità si compone di due approcci, **qualitativo** e **quantitativo**

Come si compone il progetto di valutazione dell'usabilità?

<b>CHI?</b>	Noi (3)	e	Utenti reali (24)
<b>COSA?</b>			
	Confronto Longitudinale	o	Confronto Trasversale
	Confrontare due versione diverse dello stesso sistema		Confronto tra due sistemi (siti web, app) diversi Valutazione euristica (3+3) - user test (12) - questionario (24)

Il risultato finale è una relazione

17.10.2023

I numeri indicati (6, 12, 24) sono una sorta di **lower bound** statistico per ottenere un risultato statisticamente plausibile

## Valutazione Euristica (Qualitativa)

La "metodologia" migliore per acquisire competenze in merito di valutazione dell'usabilità, è quella del miglioramento continuo **Plan-Do-Check-Act**

Spesso, in contesto di usabilità, si utilizza il termine di "**mock-up**" per indicare prototipi non funzionali sui quali si testa l'interfaccia e l'interazione in fase di Check

La valutazione cosiddetta euristica è una valutazione di tipo **qualitativo** semantica

Il primo passo per effettuare una valutazione qualitativa, è coinvolgere gli utenti, divisi in due categorie:

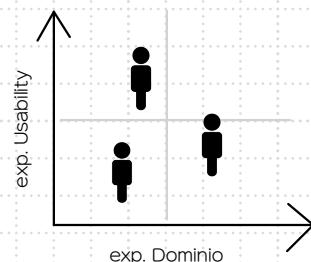
#### Esperti di dominio (del task)

#### Esperti di usabilità

Si valuta un sistema interattivo alla luce dei principi di buona progettazione (**euristiche**), in modo completo e adeguato, al fine di farsi un'idea di tutti i possibili problemi di usabilità che un sistema può presentare se usato veramente

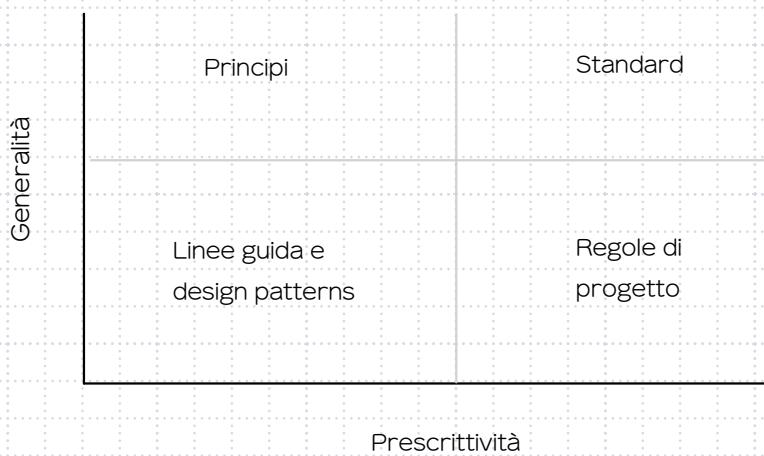
L'output di questa valutazione è un elenco di problemi di usabilità rilevati nel sistema

Nella relazione di progetto, bisogna inserire un grafico in cui collocare gli utenti coinvolti, dividendoli in zone individuate su due assi, in base all'expertise sull'usabilità (y) e sul dominio (x)



Quali sono le indicazioni per il design dell'interfaccia utente?

Prescrittività alta significa che non c'è margine di interpretazione  
Generalità significa applicabilità in contesti meno uniformi



I **principi** sono indicazioni generali da seguire per la progettazione di interfaccia usabili, basate su evidenza scientifica o consenso generale

Gli **standard** sono insiemi di regole da applicare nel progetto di una classe di sistemi, possono anche essere vincolanti. Sono di norma emessi da enti di standardizzazione

Le **linee guida e design pattern** sono raccomandazioni per il progetto di una particolare classe di sistemi; sta al progettista decidere se implementarle o meno

Le **regole di progetto** sono regole o specifiche da applicare nel progetto di un particolare sistema, definite in base ai precisi requirements del cliente; sono spesso vincolanti

Si chiamano principi euristici (o solo **euristiche**) l'insieme di concetti che fanno riferimento a strategie progettuali rivelatesi adatte e "di successo" in vari scenari applicativi; sono delle "shortcut cognitive" a cui ci si può affidare e che si possono adottare

La valutazione euristica è quindi la valutazione di usabilità svolta alla luce di un determinato insieme di euristiche per identificare soluzioni di design di successo

Si vuole valutare l'usabilità **ad alto livello**, anziché con focus stretto solo su linee guida e pattern, al fine di non creare vincoli eccessivi nei confronti di **originalità** e **innovatività**

Ci sono diversi livelli di **autorevolezza** (o forza dell'evidence) per classificare le "prove":

**Lv. A:** completamente supportate da risultati di ricerca e dati empirici

**Lv. B:** basate su pratiche generalmente accettate in modo documentato

**Lv. C:** non ben documentate, ma supportate dall'opinione di professionisti esperti

**Lv. D:** opinione individuale del progettista

Negli standard sono solitamente recepite solo indicazioni di livello A e B; i principi, le euristiche, che andremo a descrivere (tra cui i principi di **Nielsen**), si collocano tra i livelli B e C e, alcune soluzioni specifiche, anche di livello A

## Esempi di **Insiemi di Euristiche** più usate nella valutazione qualitativa dell'usabilità

### IXD Checklist

Affordance	I controlli devono essere mappati all'effetto in modo semplice e logico
Feedback	C'è una metafora chiara frapposta tra il controllo e il mondo reale
Simplicity	La funzione del controllo è facilmente determinabile a prima vista
Structure	
Consistency	Tutti gli stati elementari sono illustrati in modo chiaro e efficace
Tolerance	Fornire un feedback quando un task è completato
Accessibility	Fornire un modo semplice per eseguire un "undo"

### ISO 9241-110

7 principi del dialogo uomo - macchina

1. Adeguatezza al compito
2. Auto descrizione
3. Conformità alle aspettative dell'utente
4. Adeguatezza all'apprendimento
5. Controllabilità
6. Tolleranza verso gli errori
7. Adeguatezza alla personalizzazione

### Principi Euristici di Jacob Nielsen

Lista di principi basati su un'analisi fattoriale (categorizzazione) di 249 problemi di usabilità, arrivando poi a 10 fattori più spesso ricorrenti

Esiste una forte relazione tra questi principi e quelli descritti nell'ISO 9241

**20.10.2023**

## Risorse Necessarie

Le risorse necessarie per effettuare una valutazione euristica sono:

- Un sistema funzionante, o prototipo, o mock-up
- n valutatori (3 esperti di usabilità, 3 esperto di dominio)
- Un osservatore (quando si coinvolge un utente esperto)
- Tempo

In ambito professionale, una sessione individuale di osservazione dura tra i 30 e i 90 minuti (a seconda dei task). Tra 5 e 15 per il nostro progetto

Ogni valutatore dovrebbe riesaminare l'interfaccia più volte

Esistono due tipi di stime riguardo a task e lavori:

**Stima elapsed**, o di calendario, indica il tempo necessario per il completamento

**Stima effort**, indica il tempo di lavoro necessario **effettiva**, dipende dal numero di persona coinvolte (concetto di Man/Month)

## Pianificazione

La pianificazione di una valutazione euristica efficace è composta di diversi step; ci si mette d'accordo sui metodi di valutazione e le notazioni, si studia l'ambiente e le necessità degli utenti, e si prepara un briefing con i valutatori

Si preparano gli scenari per i valutatori; compiti e task ben definiti da svolgere. Si prepara l'ambiente di valutazione, ovvero predisporre i supporti da dare agli utenti valutatori

È importante che tutti i valutatori ricevano spiegazioni in merito ai task in modo equivalente  
L'ambiente dev'essere standard e pulito per tutti gli utenti

Dopodiché avviene la valutazione vera e propria da parte degli utenti. In seguito, si osservano le sessioni di valutazione, e segue una fase di debriefing per elencare i problemi di usabilità rilevati

È necessario analizzare i problemi riscontrati **uno per uno**, dopo questa fase è necessario avere un elenco con il numero maggiore di problemi

Non è necessario, e anzi è (generalmente) sbagliato, proporre soluzioni ai problemi già in questa fase (non sarà necessario per il progetto)

Partendo dall'elenco dei problemi, ogni problema lo si deve collegare a una o più euristiche che esso viola. Passare in rassegna tutte le euristiche per ogni problema è un approccio efficace: si crea un mapping "n a n" tra problemi e principi

L'output della valutazione è un **elenco di problemi prioritizzati**

## Prioritizzazione

È necessario, spesso tramite un questionario, coinvolgere utenti per valutare la **severità** di ognuno dei problemi individuati. Idealmente, si hanno tra le 20 e 30 valutazioni ordinali (ne saranno sufficienti 6 per il progetto)

Il fine è indicare quali sono i problemi più urgenti da risolvere

Come procedere?

1. Estrarre dal sistema di rilevazione (questionario online) la tabella delle valutazioni (i problemi indicati sulle colonne, e le valutazioni di severità, da 1 a 4, una per singolo valutatore)
2. Per ciascuna riga, adottare la strategia di competizione standard (**1224**), ovvero per ogni rater, stilare una classifica parziale dei problemi

	Prob 1	Prob 2	Prob 3		Prob 1	Prob 2	Prob 3
Valutatore 1	3	3	1	→	1	1	3
Valutatore 2	4	5	3		2	1	3

Disegnare anche una matrice che ha per righe gli indici dei valutatori e per colonne gli indici dei problemi, e colorarle le celle in base a quale valutatore ha identificato quale problema; colorare la cella con una tonalità diversa in base alla gravità dell'errore

3. Considerare due fasce: quella dei problemi a priorità più alta considera le prime n posizioni, con n circa 20% del numero totale dei problemi
4. Contare, per ogni problema, il numero di volte che esso è in prima fascia (p) e il numero di volte che è in seconda fascia (s)

	Prob 1	Prob 2	Prob 3		Prob 1	Prob 2	Prob 3
1	1	1	3		2 v 0	2 v 0	0 v 2
2	1	1	3	n = 2			

5. Assegnare il problema ad una fascia in base al confronto tra p ed s  
Se p > s, il problema è in prima fascia, altrimenti è in seconda fascia
6. Eseguire un test binomiale per capire la differenza tra p ed s è dovuta al caso.  
Stabilire così tre fasce:

Fascia alta con priorità significativa (in base al P value del test)

Fascia intermedia di non significatività (in base al P value del test)

Fascia bassa con priorità significativa (in base al P value del test)

	Prob 1	Prob 2	Prob 3		Prob 1	Prob 2	Prob 3
	20 v 6	23 v 4	2 v 25		A	A	C

7. Tutti i problemi nella fascia alta dovranno essere risolti prioritariamente, e si possono scegliere anche alcuni problemi della fascia intermedia
8. Nella tabella dei problemi, aggiungere due colonne: la fascia di priorità e la media o mediata dell'indice di severità

## Fase di Rapporto

Il rapporto finale (slideware) della valutazione euristica deve contenere:

Descrizione del design

Cosa è stato valutato, perché , coinvolgendo quante persone,considerando quali task, aerogramma di profilazione, grafico expertise...

Elenco finale dei problemi prioritizzati

Tabella dei problemi con le varie colonne, screenshot dei problemi più gravi, matrice e problemi/valutatori, distribuzioni problemi/euristiche

Considerazioni finali

Legate al numero totale stimato di problemi, alla distribuzione dei problemi, alle citazioni notevoli delle sessioni olistiche ed esperte

Allegare le n schede di analisi (3 redatte da noi esperti di usabilità, 3 redatte dagli esperti in base alle registrazioni degli altri utenti), e le registrazioni e i moduli di consenso

**26.10.2023**

### Euristiche di Nielsen

1. Far vedere lo stato del sistema (feedback)
2. Adeguaire il sistema al mondo reale (parlare il linguaggio dell'utente)
3. Controllo dell'utente e della libertà (uscire indicare chiaramente)
4. Assicurare consistenza
5. Riconoscimento piuttosto che uso della memoria (dell'utente)
6. Assicurare flessibilità ed efficienza d'uso
7. Visualizzare tutte e sole le informazioni necessarie
8. Prevenire gli errori
9. Permettere all'utente di correggere gli errori
10. Help e documentazione

Percezione Cognizione Errori

## Euristiche nel Dettaglio

### 1. Far vedere lo stato del sistema (feedback)

Il sistema deve informare continuamente l'utente di ciò che sta facendo e come sta interpretando gli input dell'utente

Il sistema deve fornire informazioni sui malfunzionamenti, fornendo plausibili ipotesi delle casue

Lo status del sistema dev'essere comunicato chiaramente e concretamente; il feedback può essere permanente o momentaneo

- Se il tempo di risposta è minore di 0.2 secondi, non serve mostrare feedback
- Se il tempo di risposta è tra i 0.2 secondi e i 3 secondi, è opportuno mostrare che c'è un'attività in corso, non è necessario dettagliare troppo
- Se il tempo di risposta è maggiore di 3 secondi, è necessario mostrare la progressione del lavoro (e.g. barra di avanzamento)

È importante mostrare informazioni utili tramite il feedback, senza un livello di dettaglio troppo elevato (il sistema è una sorta di black box); mostrare il **cambiamento** di una certa azione dell'utente è una buona idea

Un messaggio di feedback ha una certa **persistenza**, che può essere:

- Breve: un messaggio che dura il tempo necessario per essere letto (rilevato), ad esempio, che la stampante ha finito la carta
- Media: un messaggio che sta sullo schermo finché l'utente non lo riconosce esplicitamente
- Alta: un messaggio che rimane permanentemente sullo schermo, ad esempio, l'icona per un basso livello di carica

### 2. Adeguare il sistema al mondo reale (parlare il linguaggio dell'utente)

Permette all'utente di sfruttare l'esperienza e la sua familiarità per interagire col sistema

I dialoghi devono essere scritti nel **linguaggio dell'utente**, ad esempio, inserire codici o acronimi non convenzionali all'interno di messaggi è una cattiva pratica

Il discorso del linguaggio si estende anche alle **icone**, le cui interpretazioni spesso richiedono conoscenze a livello culturale

### 3. Controllo dell'utente e libertà

Quanto deve essere controllato l'utente e quanto deve essere autonomo?

Il rapporto tra questi due parametri è proporzionale, e potrebbe essere dinamico, ovvero cambiare in base al contesto

L'utente deve sentire di essere in **controllo** del dialogo, non di essere "intrappolato" dentro al sistema; è necessario introdurre sempre una **via d'uscita semplice** dalla situazione corrente

Ogni vincolo o controllo che viene inserito deve essere di chiara **comprendione**, in modo che di fronte ad un blocco, l'utente sa come procedere

Bisogna offrire sempre delle possibilità di **undo**, ovvero la possibilità di annullare un comando o il suo effetto; è una delle massime espressioni di controllo da parte dell'utente

Idealmente, l'undo deve essere illimitato, ovvero l'utente deve essere in grado di annullare più operazioni, tornando ad uno stato sempre precedente

- **Flip undo:** è possibile effettuare un undo per annullare un undo
- **Multiple-step undo:** a differenza del flip undo, questo tipo di undo non agisce su se stesso, permettendo di tornare indietro (e avanti) per un numero elevato (idealmente illimitato) di operazioni

Vie d'uscita e undo permettono all'utente di imparare, **esplorando**, il sistema

In caso di operazioni che durano troppo, ovvero formalmente più di 10 secondi, dev'esserci la possibilità da parte dell'utente di annullare o cancellare l'operazione

#### 4. Assicurare consistenza

Questa euristica è uno dei principi base dell'usabilità; qualsiasi tipo di scelta (grafica, di architettura...), va mantenuta per **tutto** il sistema, ogni pagina, ogni finestra...

La consistenza aiuta l'utente a crearsi una sua abitudine, a riconoscere gli oggetti o i messaggi del sistema

Ad esempio, cambiare la posizione o l'ordine di controlli (come dei pulsanti), è un esempio di **inconsistenza del layout**, è un grave errore di usabilità

Le **toolbars** (o **barre dei controlli**) sono solitamente un elemento critico di consistenza; Una barra principale deve persistere, anche come stile visivo, tra le varie pagine in cui porta dopo che l'utente di interagisce

Nell'ambito del gruppo di progetto e sviluppo, è importante definire **convenzioni**, di interazione e di rappresentazione, che valgano per tutti il progetto

Vuoi davvero cancellare il file?

**Si, cancella**

**No, annulla**

L'accoppiamento verde-rosso è qualcosa di desueto e consistente, non comunica bene all'utente

Vuoi davvero cancellare il file?

**Si, cancella**

**No, annulla**

L'ultima soluzione sceglie solo il rosso come accento, e comunica due cose: l'azione **cancella**, evidenziata, è quella per cui è chiesta la conferma, quella che vuole fare l'utente e, siccome è colorata di rosso, può provocare "danni"

Vuoi davvero cancellare il file?

**Annulla**

**Cancella**

Non c'è sempre una risposta giusta o sbagliata, l'importante è progettare l'accessibilità in modo che sia **empiricamente** comprensibile dagli utenti, coinvolgere persone e chiedere quale soluzione preferiscono, scegliendo poi per maggioranza

Per scegliere quindi l'opzione che "produce meno danno" tramite il feedback degli utenti, si conducono degli **A/B test**, ovvero si sceglie una porzione causare degli utenti, e gli si propone una nuova modifica o funzione per un tempo limitato

## 5. Riconoscimento piuttosto che uso della memoria

"L'utente passa la maggior parte del suo tempo su un'altra applicazione"

È conveniente sfruttare le conoscenze dell'utente e giocare sulla familiarità dell'interfaccia; l'uso di **comandi generici** aiutano il trasferimento di informazione da un sistema ad un altro

Bisogna fare attenzione all'**information overload**, ovvero non esagerare con le informazioni date all'utente

## 6. Assicurare flessibilità ed efficienza

Spesso, questi due concetti si trovano in contrapposizione tra di loro; è necessario prendere atto delle differenze tra i tipi di utente che utilizzano il sistema

L'utente **esperto** deve poter svolgere velocemente le operazioni più frequenti

**Shortcut** da tastiera sono **acceleratori**; altri esempi sono i **bookmark**, o **breadcrumb trails**, **type-ahead**, definizione di **macro** o **template**

## 7. Visualizzare tutte e sole le informazioni necessarie

È impossibile capire come implementare questo principio senza il coinvolgimento degli utenti; è necessario comprendere quali informazioni sono davvero necessarie, e quali invece risulterebbero un information overload

Una pagina deve essere **semplice** e **gradevole**, e rispecchiare nel modo più naturale possibile i compiti dell'utente

Bisognerebbe presentare solo l'informazione:

- **Che** l'utente vuole
- **Quando** l'utente la vuole
- **Come** l'utente la vuole
- **Dove** l'utente la vuole

**Raggruppare** le informazioni utilizzate insieme o appartenenti a domini simili è una buona tecnica per rendere la pagina più chiara e comprensibile

Il bells and whistles bias è quella credenza errata per cui più opzioni si mettono a disposizione per l'uso, più l'utente è soddisfatto

Offrire diversi **modi utente** può essere una più funzionale alternativa: definire diverse modalità (esperto, novizio, operatore...) in cui l'utente può interagire

Il fenomeno del **low use** è quel fenomeno che avviene quando una certa funzionalità (o un insieme di esse) non viene utilizzata o viene utilizzata poco perché difficile da usare

Riguarda il rapporto tra quelle "funzioni che non sono note, o sono note ma usate praticamente mai" e quelle usate spesso, si considera solo quelle note e si calcola uno **Use Index**:

In base al valore ottenuto si comprende se il sistema è o meno soggetto a low use

Aiuta a capire quanto potenziale della nostra applicazione viene usato dagli utenti, e ha priorizzato la manutenzione correttiva ed evolutiva; si punta sulle funzionalità più usate e si lasciano indietro quelle meno usate

La grafica, quindi il layout e il colore, l'estetica, è un aspetto importante;

- Le informazioni più rilevanti devono essere "messe prima"
- Caratteri maiuscoli o elementi lampeggianti rendono la lettura più lenta, usarli solo se necessario
- Non usare più di 5-7 colori, evitando colori brillanti per lo sfondo
- Assicurarsi che gli elementi dell'interfaccia si distinguono anche senza colori
- Non dedicare troppo spazio a "metadati" (del venditore, numero di versione)

In generale, la velocità di lettura media è di circa 200-300 wpm per la lettura generica, ma tale velocità dipende anche dal task, dall'illuminazione, dai font, dal livello di scolarità

Per calcolare la persistenza dei messaggi di allerta o help, considerare 200 wpm

**27.10.2023**

## 8. Prevenire gli errori

Cercare di progettare un sistema in grado di prevenire situazioni in cui l'utente può effettuare errori; individuare le situazioni di possibile errore

Creare **meccanismi di prevenzione**, come ad esempio, prevenire una digitazione errata facendo scegliere l'opzione all'utente da un menù a tendina

Si possono ridurre errori tramite **richieste di conferma** di un'azione da parte dell'utente, per quelle azioni che producono effetti importanti

Evitare di usare comandi troppo simili con significati ed effetti diversi

Creare **meccanismi di comunicazione utente - progettista**, ovvero creare un canale di comunicazione (anche non diretta e asincrona) tramite cui l'utente possa informare il progettista di errori che riscontra durante l'uso (ad esempio, log molto ricchi per seguire il workflow dell'utente)

Ci sono varie tecniche di prevenzione:

- Diversificare le azioni dell'utente
- Implementare amodalità
- Vincolare le funzioni
- Avvertimenti e richieste di conferma
- Impostare default inoffensivi
- Bypass sicuri (workaround previsti e prestabiliti)

Si possono anche distinguere tre livelli di allerta:

- |  |  |
|--|--|
| 1. Tieni presente che... ( <b>Note</b> )         | Files cannot contain a ":" , old name restored |
| 2. È proprio quello che vuoi? ( <b>Caution</b> ) | Do you really want to delete this file?        |
| 3. Fermo! ( <b>Stop</b> )                        | The Trash cannot be moved off the desktop      |

## 9. Permettere all'utente di correggere gli errori (e non solo di rilevarli)

Utilizzare **buoni messaggi d'errore**, scritti in linguaggio chiaro e con indicazioni precise su come risolvere il problema, **recoverability**, ovvero la possibilità di raggiungere il risultato desiderato anche se si è commesso un errore

I messaggi d'errore devono essere precisi e costruttivi;

Invece di dire:

"Non posso aprire questo file"

Dire:

"Non posso aprire il documento 'Capitolo 5' perché non è sul disco"

Evitare anche messaggi aggressivi o intimidatori, essere gentili

Si indicano diversi aspetti per la segnalazione di un errore:

- **Alert** "Qualcosa non va"
- **Identify** "Questa cosa non funziona"
- **Direct** "Risolvere in questo modo"

Tutti e tre questi aspetti dovrebbero fare parte di un messaggio d'errore efficace

Bisognerebbe evitare di condensare più errori nello stesso messaggio: meglio tante finestre (modali) di errore, una per ogni errore

## 10. Fornire aiuto e documentazione

La cosa migliore sarebbe fare un sistema talmente tanto facile da usare che non ha bisogno di un manuale

Non è però del tutto possibile, quindi è ancora necessario prevedere manuali e help (anche tramite diversi canali di comunicazione, come video); **tooltip** e aiuti **context-aware** sono un'ottima idea di implementazione

## **Valutazione Quantitativa**

Un sistema interattivo è di successo se è:

**Utile**: deve fare quello che viene richiesto, risultare importante per l'utente

**Usabile**: deve consentire di fare questo che viene richiesto in modo naturale e intuitivo

**Usato**: deve rendere le persone desiderose di usarlo, dev'essere interessante, divertente, piacevole (**UX** user experience e **IxD** interaction design)

## **Usabilità**

Secondo lo standard ISO 9241, l'usabilità è definita come il grado con cui un prodotto può essere usato da determinati utenti per raggiungere determinati obiettivi con efficacia, efficienza, soddisfazione in un determinato contesto d'uso

Si lavora in termini **probabilistici** (per questo usa il termine "può"), in quanto la variabilità tra tipologia di utente, di obiettivo e di contesto è molto elevata

L'efficienza è il **rapporto** tra il risultato raggiunto e il costo, le risorse utilizzate per raggiungerlo; un rapporto tra le risorse utili e quelle effettivamente usate.

"Effectiveness is doing the thing right, efficiency is doing these things right"

La soddisfazione è la misura di quanto è **contento** l'utente, se l'esperienza d'uso è stata piacevole (**UX**, ciò che una persona prova quando usa il sistema)

L'esperienza d'uso ha una natura **soggettiva**, perché riguarda i pensieri e le sensazioni del singolo utente, e non è un parametro certo e assoluto come l'efficienza

Il contesto d'uso è una variabile di grande impatto, ci possono essere vari contesti; primo utilizzo, in caso di emergenza...

Non ha senso parlare di **usabilità** in modo assoluto, di un sistema in sè, ma piuttosto in un contesto d'uso e con un determinato utente; ha senso parlare di **potenziale usabilità**

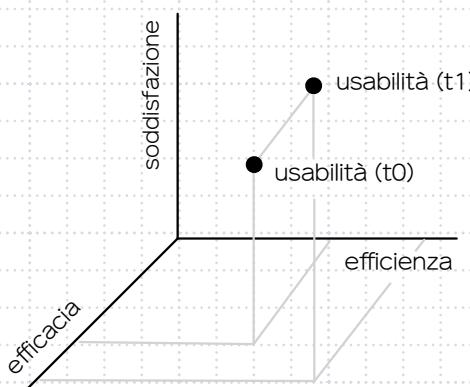
Infine, si parla di **grado** di usabilità, definendo delle metriche per **misurare** l'usabilità

### Misura

Risultato di una misurazione;  
Associazione empirica ed  
oggettiva di un valore ad  
un'entità per caratterizzarne  
un attributo specifico

### Metrica

Comprende la misurazione quantitativa, e  
definisce anche le procedure, le modalità e  
le regole di misurazione; stabilisce: le  
entità e gli attributi da misurare, l'unità di  
misura, e una procedura di misura



Una differenza nell'usabilità può  
essere praticamente e/o  
statisticamente significativa; una  
differenza praticamente significativa  
è quella che fa davvero differenza  
a livello pratico (di guadagno);  
statisticamente significativa  
significa "non dovuta al caso"

