

Appunti di Programmazione di Interfacce

Simone Pepi¹ e Francesco Iannelli²

¹Stesura appunti

²Riscrittura in L^AT_EX

a.a. 2019/2020 Prof. Mazzei

Capitolo 1

Introduzione

Che cos'è il design?

Il design è la **pianificazione o la specifica per la costruzione di un oggetto o sistema, o per l'implementazione di un'attività o un processo**. Si trova quindi agli antipodi della scomposizione del problema in sottopassaggi, cioè del pensiero computazionale.

Il design parte dalla base del problema e **identifica soluzioni per la causa del problema**. Famosa è la frase: *"Se vogliamo che agli utenti piaccia il nostro software, dobbiamo progettare le applicazioni come se fossero persone con cui ci piacerebbe uscire."*

Due delle caratteristiche più importanti di una buona progettazione sono **visibilità** e **comprendibilità**.

- **Visibilità:** è possibile indovinare quali azioni sono possibili e come eseguirle?
- **Comprendibilità:** cosa significa tutto questo? Come va usato? Cosa significano tutti i vari comandi?

Nei dispositivi complessi la visibilità e la complessità richiedono l'uso di manuali d'istruzioni, ma questo lo accettiamo solo se il dispositivo è davvero complesso, ma dovrebbe essere del tutto superfluo per le cose semplici.

1.1 XX Design

Fin dai primi tempi dell'industria del design, la parola *design* normalmente farebbe pensare a qualcosa riguardante la grafica. Il lettore potrebbe sentirsi confuso riguardo a tutte le sfumature di significato che assume questa parola nel mondo del lavoro.

Con l'evoluzione della tecnologia anche l'industria dei media si è evoluta, dalla stampa ai media sul web, mobile e software, così come l'industria del design.

I settori principali della progettazione che andremo a toccare sono: il **design industriale**, il **design dell'interazione** e il **design dell'esperienza utente**. Nessuno di essi è rigidamente definito ma è diverso il punto focale di ognuno:

- **Design industriale:** creazione e sviluppo sia di concetti che di specifiche per ottimizzare la funzionalità, il valore e l'aspetto di prodotti e sistemi, con reciproco vantaggio per gli utenti e i produttori
- **Design dell'interazione:** l'attenzione è concentrata sul modo in cui le persone interagiscono con la tecnologia; lo scopo è migliorare la loro comprensione di ciò che si può fare, ciò che succede e ciò che è appena successo, basandosi su principi psicologici, tecnici ed estetici.

- **Design dell'esperienza utente:** progettazione di prodotti, processi, servizi, eventi e ambienti, mirando soprattutto alla qualità e alla piacevolezza dell'esperienza complessiva.

Per rispondere chiaramente a qualsiasi domanda, dobbiamo innanzitutto illustrare le differenze tra **Graphic Design**, **User Experience Design (UX Design)**, **User Interface Design (UI Design)**.

1.2 UX Designer

La **User Experience** è il modo in cui le persone si sentono **a livello psicologico** quando usano un prodotto. Ogni prodotto utilizzato da qualcuno ha una user experience: giornali, bottiglie di ketchup, poltrone reclinabili, maglioni di cardigan ecc...

Quindi l' **UX design** si occupa di come il prodotto viene recepito, la sensazione che dà, ed è molto diverso dal tradizionale design grafico.

I progettisti UX lavorano in modo da conferire al prodotto la migliore esperienza d'uso possibile, in modo da trasmettere, all'utente che ne farà uso, una sensazione di soddisfazione e non di frustrazione. A tal fine vengono usati una varietà di strumenti tra i quali: analisi competitiva, interviste e sondaggi, il tutto per arrivare a costruire tipologie di *personas* tra gli utenti del prodotto.

Non si può progettare la User Experience, ma si può progettare per la User Experience, ecco perché, per soddisfare appieno gli utenti che useranno il prodotto, è importante comprendere quali sono le loro necessità.

1.3 UI Designer

Dallo studio della UX si crea un abbozzo dell'interfaccia. Non si crea subito il wireframe finale, ma si parte da un analisi dei casi di studio. Esistono più tipi di casi di studio ed ognuno è specifico per delle personas, infatti, personas differenti hanno capacità differenti.

L'UI Design è un procedimento differente dal front-end developing: la materia progetta le guidelines che istruiscono il developer su come creare al meglio una UI.

Possiamo considerare la UI Design come **sotto area** della UX Desing.

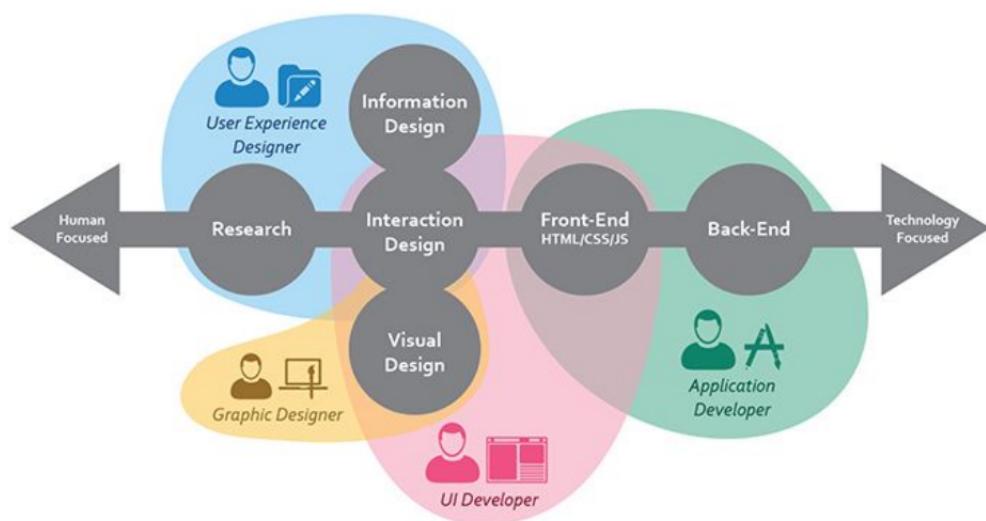


Figura 1.1: Le varie aree del design.

Capitolo 2

User Interface

Quando parliamo di **User Interface (UI)**, in italiano Interfaccia Utente, parliamo dello spazio di un sistema dove avviene l'interazione uomo-macchina: il monitor, il mouse, le casse audio e quant'altro.

L'obiettivo di questa interazione è far sì che l'utente possa controllare e far funzionare la macchina in modo efficace, mentre la stessa macchina reagisce simultaneamente fornendo informazioni che aiutano il suo processo decisionale tramite **feedback**.

In generale, l'obiettivo dell' UI design è quello di produrre una UI che renda facile, efficiente e divertente l'uso di una macchina o di un programma, in modo da massimizzare la User Experience dell'utente finale.

Il termine **user-friendly** non può essere omesso in questa trattazione: tra un'app facile e piacevole da utilizzare e una solo facile da usare, l'utente medio preferirà sempre la prima.

Le interfacce sono strutturate in uno o più layer. L' **HID o Human Interface Device** è la periferica con cui l'utente interagisce con il sistema; come ad esempio mouse, monitor, gamepad, ecc...

Lo **HMI o Human Machine Interface** è un concetto che astrae dall' HID, con HMI, infatti, ci si riferisce allo strato che separa un essere umano che sta utilizzando una macchina dalla macchina stessa. Un device che implementa un HMI è un HID. Quando la macchina in questione è un computer, HMI diventa **HCI o Human Computer Interface**.

2.1 Diversi tipi di interfacce

Una persona umana possiede cinque differenti sensi che possono essere "mappati" in cinque categorie di interfacce possibili, più una categoria introdotta con l'introduzione di visori e giroscopi:

- **tactile UI** (touch)
- **visual UI** (sight)
- **auditory UI** (sound)
- **olfactory UI** (smell)
- **equilibrial UI** (balance)
- **gustatory UI** (taste)

La composizione di più UI prende il nome di **CUI (Composite User Interface)**. Le più comuni CUI sono le **GUI o Graphical User Interface**, le quali sono composte da interfacce grafiche e tattili. Se aggiungiamo anche il suono diventano **MUI (Multimedia User Interface)**.

È bene sottolineare che **aggiungere più interfacce per poter interagire con una macchina utilizzando più sensi non è sempre una buona idea**. Giusto per fare un esempio, prendiamo in esame i video di Facebook: i video venivano riprodotti con l'audio attivo, ma gli ingegneri di Facebook si sono accorti che la maggioranza delle persone che visualizzavano i video, mutavano immediatamente il suono per varie ragioni (e.g. privacy o utilizzo di Facebook in momenti non opportuni), quindi hanno ben pensato di far partire la riproduzione automatica dei video con il suono mutato e introducendo i sottotitoli per le parti del video parlate. Questo oltre ad essere un ottimo esempio di MUI riprogettata in GUI è anche un esempio di tecnica ideata per le utenti disabili e riusata per poter far fruire il prodotto a quelle personas che lo utilizzano in momenti in cui non possono usufruire dell'audio.

2.2 Categorizzare le CUI

Le CUI possono essere categorizzate in tre diverse macrocategorie:

- **Standard:** usano dispositivi standard come tastiere, mouse e monitor
- **Virtual:** schermano il mondo reale e creano un mondo virtuale. Tipicamente utilizzano dei caschi VR.
- **Augmented:** non schermano il mondo reale e erogano contenuti non completamente digitali che prendono forma nella realtà esterna che circonda l'utente, appunto espandendola.

Quando un'interfaccia utente interagisce con tutti i sensi umani viene chiamata **Qualia Interface**, secondo la **teoria Qualia**.

Le CUI possono essere anche **classificate per il numero di sensi** con cui esse interagiscono. Ad esempio, lo *Smell-O-Vision* è una CUI standard 3S (3 sensi) con display, suono e odori. Se si aggiungesse un quarto senso diventerebbe un 4S, si pensi ad esempio alle poltrone dei famosi cinema 4D.



Figura 2.1: Virtual reality



Figura 2.2: Augmented reality.

Capitolo 3

Good and Bad Design

Il buon design valido sempre e per tutti non esiste, poiché si fa design per la user experience di determinate personas. Però possiamo dare due caratteristiche importanti sui cui misurare un buon design:

- **Discoverability:** è la capacità innata di un sistema di veicolare i possibili usi e di comunicare come si usa. Non è detto che una volta capito cosa si può fare si riesca a farlo. Per avere una buona discoverability si usa tipicamente la **visibilità**: un rubinetto con i pomelli bene in vista incrementa la discoverability. In un software tale lavoro è svolto dai pulsanti.
- **Understanding:** è la capacità del prodotto di farsi usare correttamente dall'utente.



Figura 3.1: Discoverability and visibility.



Figura 3.2: Understanding.

3.1 Design of Useful Things

Quando le cose vanno bene, si dimenticano subito!

Questo perché nella psicologia umana le cose devono andare bene per definizione. Quando qualcosa va storto invece, l'amigdala crea un ricordo con un peso molto maggiore rispetto a un ricordo di una esperienza piacevole.

Il design deve quindi preoccuparsi di come funzionano le cose, come vengono controllate e della natura delle interazioni. Quando la progettazione è fatta bene, crea prodotti piacevoli e brillanti, quando è fatta mala, i prodotti sono inutilizzabili e ciò porta a una notevole frustrazione e irritazione.

Le macchine sono concepite, progettate e costruite da esseri umani. Al nostro confronto sono **assai limitate**: non conservano quella ricca storia di esperienze comuni che ci permettono di interagire grazie a un patrimonio collettivo di conoscenze.

Le macchine seguono di solito regole di comportamento rigide, piuttosto semplici. Se sbagliamo nel seguirle, anche di poco, la macchina fa quello che le diciamo, per quanto insensato e illogico sia. Noi esseri umani siamo creativi, dotati di immaginazione e pieni di buon senso, ovvero un abbondante patrimonio di sapere accumulato in anni di esperienza. Le macchine, però, ci obbligano a una grande precisione, cosa a cui non siamo avvezzi. Le macchine non hanno né flessibilità né buon senso e, spesso, gran parte delle regole seguite da una macchina è nota solo alla macchina stessa e ai suoi progettisti.

Quando non si eseguono queste sue regole segrete e bizzarre, e la macchina fa la **cosa sbagliata**, la colpa viene scaricata su chi la manovra, accusato di non capirla e di non seguirne i rigidi protocolli. Con gli oggetti di uso comune, il risultato è la frustrazione, con i dispositivi complessi o processi industriali e commerciali, le conseguenze possono essere incidenti anche mortali.

È tempo di **ribaltare la situazione, di accusare le macchine e la loro progettazione**. La colpa è delle macchine e di chi le ha progettate: sta a loro capire le persone, non a noi capire i loro dettami arbitrari e insensati.

Le ragioni delle carenze nell'interazione uomo-macchina sono numerose.

Alcune nascono dai limiti della tecnologia attuale, altre da limitazioni intenzionali dei progettisti, spesso per abbassare i costi di produzione. Ma la maggior parte dei problemi deriva dalla totale incomprensione dei principi di design necessari per un'efficiente interazione uomo-macchina. Perché questa deficienza? Perché la progettazione è opera per lo più di ingegneri esperti di ingegneria ma non di psicologia, quindi limitati nella comprensione delle persone.

"Siamo uomini anche noi" pensano, *"quindi siamo in grado di capire i nostri simili"*.

I tecnici fanno l'errore di pensare che sia sufficiente la spiegazione logica: *"Basterebbe che leggessero le istruzioni e andrebbe tutto bene"*.

Gli ingegneri sono formati a un tipo di pensiero logico, di conseguenza finiscono per credere che tutti debbano pensare in quel modo e progettano le loro macchine di conseguenza.

**Dobbiamo accettare il comportamento umano
per quello che è, e non come vorremmo che
fosse.**

Capitolo 4

Human Centered Design

Le persone sono frustate dalla complessità degli oggetti quotidiani. Dalla complessità sempre maggiore del cruscotto dell'auto, alla crescente automazione della casa, con le sue reti interne. La proliferazione di sistemi complessi per il tempo libero e la comunicazione (e.g. video, audio, giochi elettronici) e le cucine sempre più tecnologiche; la vita di tutti i giorni sembra a volte una battaglia infinita contro la confusione, gli errori continui, la frustrazione, e un ciclo interminabile di aggiornamento e manutenzione degli apparecchi.

La soluzione è il **Design Antropocentrico** o **Human Centered Desing** o **HCD**, un'impostazione che parte dai bisogni, dalle capacità e dai comportamenti umani, adattando la progettazione a quei bisogni, quelle capacità e quei comportamenti. Lo HCD è un approccio di design specificamente orientato allo sviluppo di sistemi interattivi con l'obiettivo di produrre sistemi utili, altamente usabili e che si **focalizzano sull'utente**.

Il metodo è orientato all'efficienza ed all'efficacia, per aumentare la soddisfazione dell'utente ed evitare il più possibile gli effetti negativi.

Prima l'utente, poi le features! Lo HCD mette i bisogni, comportamenti e capacità umane prima di tutto, e progetta in funzione di esse.

Il problema principale delle UI è la comunicazione, in particolare la comunicazione dalla macchina verso la persona; **una buona interfaccia sa comunicare con l'utente**.

Progettare interfacce che funzionano fintanto che le cose vanno bene è relativamente facile, ma **la comunicazione è ancora più importante quando le cose non vanno bene**. È qui che i progettisti devono concentrare l'attenzione, sui casi in cui le cose vanno storte, non su quelli in cui le cose funzionano secondo i piani. Si focalizza l'attenzione soprattutto nel **comunicare ciò che è andato storto**: bisogna guidare l'utente frustato alla risoluzione del problema poiché, in caso di **risoluzione**, proverà una **sensazione positiva** di successo per aver capito cosa non funzionava e per aver risolto il problema. Ciò crea **empatia con il sistema**.

Evitare quindi la frustrazione e aiutare a risolvere quando insorge un problema sono i concetti chiave dello HCD. Lo HCD è una filosofia di design che parte dalla **comprensione delle persone e dei bisogni che intende soddisfare**. Questa comprensione deriva dall'osservazione e dallo studio delle persone che spesso sono inconsapevoli dei loro veri bisogni e magari nemmeno delle difficoltà che incontreranno.

Per capire l'utente serve studio e osservazione della persona stessa. Non è sempre possibile tale osservazione, per cui versioni alfa e beta di un certo sistema non servono solo a fare debugging, ma servono anche per capire che cosa fa e come si comporta l'utente: diventa utile avere statistiche sull'utilizzo effettivo del sistema (e.g. quanti click su un determinato pulsante, quante volte una determinata procedura, ecc...)

Ottenere le specifiche dello HCD è quindi una delle parti più difficili del design, al punto che il principio è quello di evitare di specificare il più al lungo possibile e procedere con ripetute approssimazioni: si esegue una specifica ad alto livello, se ne implementa una parte, si testa sull'utente finale e tramite il suo feedback, modifico la parte implementata e testo di nuovo. Quando si ritiene buono ciò che è stato implementato si congela e si passa ad implementare una nuova parte.

Il ruolo dello HCD nel design	
Experience design	Area di focus
Industrial design	Area di focus
Interaction design	Area di focus
Human Centered Design	Il processo che assicura che la progettazione incontra i bisogni e le capacità degli utenti che useranno il sistema

Possiamo progettare per il design industriale, per l'interazione e per l'esperienza utente. Lo **HCD non è un'area di focus ma è un metodo**.

4.1 Desing Thinking vs HCD

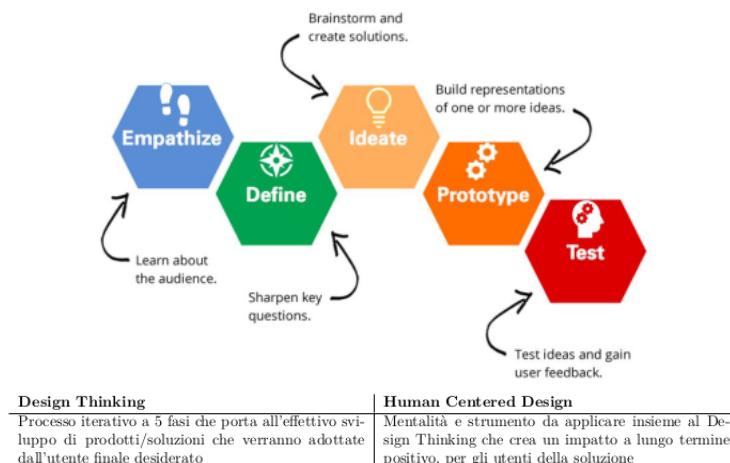
Insieme al termine HCD, spesso si può vedere il termine **Design Thinking**. I due termini vengono da scuole di pensiero molto valide ma con visioni diverse.

Il Design Thinking segue il filone Stanford (dove è nato): è un **processo di desing** con cui progettare nuovi prodotti che verranno effettivamente adottati dalle persone. Come processo è più vicino alla **disruptive innovation** che all'antropocentricità.

Si suddivide in cinque fasi iterative:

- **Empathize**: studiare il proprio pubblico, progettare il prodotto in modo che stabilisca un collegamento empatico con l'utente.
- **Define**: delineare meglio le domande chiave, cioè quali sono i bisogni a cui assolvere.
- **Ideate**: brainstorming, creare soluzioni.
- **Prototype**: costruire una o più idee.
- **Test**: testare le idee e ricevere un feedback.

Lo HCD è un mindset che viene sovrapposto al Design Thinking: identificato il modello di business, si può applicare lo HCD per assicurare che il prodotto soddisfi effettivamente le esigenze delle persone che lo andranno ad utilizzare.



Capitolo 5

Principi Fondamentali dell'Interazione

I bravi designer producono esperienze piacevoli! Ai tecnici non piace molto la parola *esperienza* poiché troppo soggettiva. Ma se si interrogasse un ingegnere sulla sua automobile preferita descriverà modello e finiture, la sensazione di potenza nell'accelerazione, la maneggevolezza del cambio e dello sterzo, ecc; queste sono esperienze.

L'esperienza è cruciale poiché determina la tonalità del ricordo che conserviamo delle interazioni con gli oggetti.

Quando la tecnologia si comporta in maniera inaspettata, proviamo confusione, frustrazione e rabbia: **emozioni negative**. Se invece comprendiamo il comportamento della tecnologia, abbiamo una sensazione di controllo, bravura e persino orgoglio: **emozioni positive**.

Cognizione ed emozione sono profondamente legate: se non mettiamo l'utente in un mood positivo farà più fatica ad apprendere l'interfaccia; più l'utente è arrabbiato e frustrato meno è predisposto a comprendere e riutilizzare il prodotto.

La **visibilità o discoverability** di un prodotto è il grado di facilità con cui un utente **scopre cosa fa, come funziona e che tipo di azione è possibile**. Tale visibilità è il risultato dell'applicazione di cinque concetti psicologici fondamentali: **affordance, significante, vincolo, mapping e feedback**. C'è anche un sesto principio, forse il più importante di tutti: **il modello concettuale del sistema**. Analizziamoli passo passo.

5.1 Affordance

Il termine affordance, letteralmente *invito*, indica la relazione fra un oggetto fisico e una persona, cioè la relazione fra le proprietà dell'oggetto e la capacità dell'utente di determinare in che modo tale oggetto può essere usato.

Una sedia appare fatta apposta per sostenere qualcosa quindi "invita" alla seduta. La maggior parte delle sedie è abbastanza leggera da poter essere sollevata e spostata da una singola persona ("invita", "permette" al trasporto), ma qualcuna è così pesante da richiedere l'intervento di più persone. Se però un certo gruppo di individui non ha la forza di sollevare una sedia, per loro la sedia non presenta l'affordance "sollevamento e trasporto".

Una affordance **non è una proprietà ma è una relazione tra un oggetto e una persona**, dipende quindi dalle proprietà sia dell'oggetto che della persona.

Si può anche parlare di **anti-affordance** nel concetto di **prevenzione dell'interazione**. Un ottimo esempio sono gli spunzoni per evitare che piccioni o altri tipi di volatili si posino in un cornicione: prevengono l'affordance di sedersi che il cornicione ha verso i piccioni. Le affordance e le anti-affordance **devono essere discoverable e perceivable**.

Questo fatto non è scontato: il vetro, famoso per la sua relativa invisibilità, occulta l'anti-affordance di precludere il passaggio.

Se uno di questi inviti o impedimenti all'uso non è percepibile c'è bisogno di qualche mezzo per segnalarne la presenza: il significante.

N.B. È assolutamente sbagliato dire "metto un affordance". Posso dire "metto un significante" ma solo se ho un'affordance.

5.2 Significante

I progettisti hanno dei problemi pratici: hanno bisogno di sapere come rendere comprensibili gli oggetti che creano. Lavorando sulla grafica degli schermi elettronici, dovevano trovare il modo di indicare quali parti potevano essere sfiorate, battute, scivolate in su o in giù o di lato, azioni che si potevano eseguire con le dite, con lo stilo o con il mouse.

Un significante è quindi **un modo per indicare dove effettuare un'azione**, dato un'affordance che determina quali azioni sono possibili.



- **Affordance:** cosa posso fare, quale azione posso compiere.
- **Signifier:** dove poso fare l'azione.

Molto spesso i significanti **sono indispensabili** poiché la maggior parte delle affordance sono invisibili. Per fare un esempio basti pensare alle porte scorrevoli: se i cardini non sono visibili, quando si vede la maniglia la prima azione che una persona tenta di fare è quella di spingere o tirare la porta, ma essa non si muoverà; è quindi necessario mettere un significante (e.g. un cartello o una scritta) che indica quale azione è necessaria per poter aprire la porta.

I significanti posso essere:

- **Voluti o intenzionali:** come un'etichetta, una stringa, un'icona.
- **Accidentali o non intenzionali:** come ad esempio un sentiero tracciato da persone che camminano attraverso un campo o delle persone in fila alla stazione.

Nel design i **significanti sono molto più importanti delle affordance**, perchè comunicano come usare il prodotto o l'interfaccia. Ma come si può associare l'affordance e il significante ad azioni reali? Nella maggior parte dei casi tramite **convenzioni**. La comprensione di un'affordance percepita è dovuta anche alle convenzioni culturali.



Figura 5.1: La scritta pull è un significante, data l'affordance della porta di essere spinta o tirata.

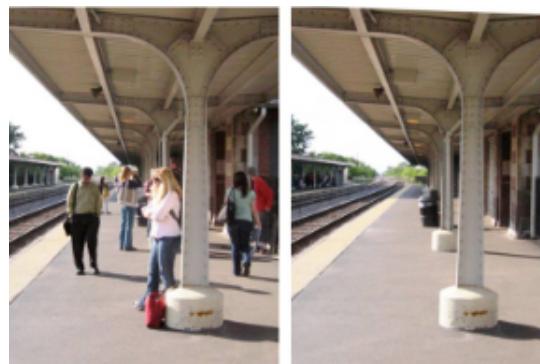


Figura 5.2: Le persone che aspettano il treno sono un esempio di significante sociale.

5.3 Mapping

Mapping è un termine tecnico, ripreso dalla matematica, che indica la relazione fra gli elementi di due insiemi.

Il concetto di mapping è di grande importanza nel progettare le interfacce e stabilire i significanti. La disposizione dei significanti può comunicare di più circa l'interfaccia e circa le sue funzionalità. Infatti quando il mapping usa una corrispondenza spaziale fra la collocazione dei comandi e quella dei dispositivi comandati, è facile capire come usarli.

Il modo migliore per applicare il mapping è quello **naturale**, perché è un'attività in cui il nostro cervello è molto bravo, i bambini imparano a fare mapping fin dai primi anni di vita. È da tenere presente che il concetto di **naturale** è ben diverso dal concetto di **universale**, poiché ci possono essere molti mapping che sembrano "naturali" ma sono specifici a una cerchia di culture.

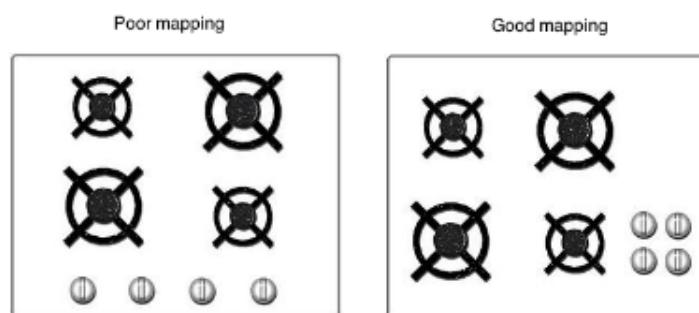


Figura 5.3: Mapping cattivo e mapping buono.

5.4 Feedback

Il feedback è la comunicazione del risultato di un'azione, di una risposta dell'interfaccia verso l'utente.

Il feedback **dove essere immediato**, anche un ritardo di un decimo di secondo può essere troppo, se il ritardo è troppo lungo l'utente potrebbe rinunciare all'attività che stava compiendo con quel prodotto e passare ad altro o addirittura non riuscire a comprendere l'origine del feedback.

Deve essere informativo, non deve portare con se troppa informazione, ma deve assolvere al proprio obiettivo, deve far capire che un'azione è in corso o che è stato prodotto il risultato che ci aspettiamo. Un scarso feedback può essere peggio di nessun feedback, perché distrae, crea confusione e quindi frustrazione da parte dell'utente.

Altro fattore importante è la **semplicità**, ovvero un feedback non deve essere pendente: troppi annunci o segnali portano le persone ad ignorarli in modo da perdere anche quei feedback cruciali e importanti. Il feedback deve essere **essenziale** e mantenere l'ambiente calmo e tranquillo.

5.5 Modello concettuale

Un modello concettuale è una descrizione altamente semplificata delle funzionalità di un sistema; non deve essere completa o accurata ma utile. I file, le cartelle e le icone che vediamo sullo schermo del computer ci aiutano a creare un modello concettuale dei dati in memoria o delle applicazioni disponibili. In realtà il computer non contiene fascicoli o cartelle: esse sono solo concettualizzazioni ideate per facilitarne l'uso.

I modelli semplificati sono preziosi e utili fintanto che le ipotesi che li supportano sono vere.

Nel Cloud Storage Sync i file sembrano essere sul dispositivo, ma in molti casi il materiale è nel cloud. Il modello concettuale è quello di un archivio disponibile sui dispositivi degli utenti. Questo modello semplificato è utile per il normale utilizzo, ma se il collegamento dei servizi si interrompe, può nascere confusione: l'informazione è sempre presente sullo schermo, ma non possiamo più salvarla o recuperare altri dati, cosa inspiegabile in relazione al modello concettuale precedentemente citato.

Il modello concettuale è **come il designer vuole che l'utente percepisca il prodotto**; sarebbe l'ambizione di progettare e comprendere la UX.

Una volta che i progettisti hanno pensato e progettato il modello concettuale si implementa l'interfaccia, in modo che il modello concettuale venga veicolato all'utente tramite affordance, significanti e mapping presenti su essa.

Quando una persona si interfaccia con il sistema o il prodotto sviluppa un suo modello mentale. Un **modello mentale** è un modello concettuale nella mente dell'utente che rappresenta il modo in cui, secondo lui, funzionano le cose. Non solo persone diverse possono avere modelli mentali diversi dello stesso oggetto, ma la stessa persona può avere molteplici modelli, pertinenti ciascuno a un aspetto diverso del suo funzionamento, e persino contraddittori gli uni con gli altri.

Più è grande la differenza tra il modello mentale e quello concettuale, più l'utente farà fatica ad usare il sistema.

L'ideale è che l'utente apprenda un modello concettuale giusto **direttamente dal device che utilizza**, senza andare a leggere manuali o istruzioni o, peggio ancora, trasmessi da persona a persona. La comprensione di un dispositivo tramite passaparola porta all'effetto del "telefono senza fili": l'interpretazione cambia da persona a persona. Per questo vi è necessità che il modello concettuale trasmesso dal prodotto sia pressoché univoco con quello mentale che l'utente si è fatto.

In questo contesto vale l'affermazione *less is more* secondo cui se una feature è difficile da veicolare allora è meglio non implementarla.

5.6 Immagine di Sistema

Le persone si creano di continuo modelli mentali di sé, degli altri, dell'ambiente degli oggetti con cui interagiscono: modelli concettuali formati attraverso l'esperienza, l'addestramento e l'istruzione.

Questi modelli ci servono da guida per realizzare i nostri scopi e comprendere il mondo in cui viviamo.

Come ci formiamo un modello concettuale adeguato dei dispositivi che utilizziamo? Non potendo parlare con il progettista, ci basiamo su tutta l'informazione accessibile: l'aspetto dell'apparecchio, cosa abbiamo imparato dall'uso di oggetti simili in passato, cosa ci dicono le pubblicità, i venditori, i pieghevoli illustrativi, il sito web e il libretto di istruzioni. **L'insieme di tutta questa informazione è l'immagine di sistema.**



Come illustrato nella figura, il progettista e l'utilizzatore finale del prodotto costituiscono i vertici scollegati di un triangolo. Un vertice del triangolo è occupato dal modello concettuale del progettista, **cioè dalla sua concezione del prodotto in questione.**

Una volta commercializzato, il prodotto si stacca dal progettista: lo vediamo isolato al secondo vertice del triangolo.

L'immagine di sistema è tutto ciò che si percepisce dalla struttura fisica prodotta (completa di documentazione, istruzioni, significanti e ogni informazione accessibile dal sito web o dal servizio di assistenza clienti).

Il modello concettuale dell'utente deriva dall'immagine di sistema, mediante l'interazione con prodotto, letture, ricerca online e manuali. Il progettista si aspetta che il modello concettuale dell'utente coincida col suo, ma, non essendoci comunicazione diretta fra lui e l'utente, tutto il peso della comunicazione grava sull'immagine di sistema.

Questo spiega perché la comunicazione è un aspetto importante del buon design. **Per quanto sia geniale il prodotto, se la gente non riesce ad usarlo l'accoglienza sarà cattiva.** Tocca al progettista fornire l'informazione adeguata a renderlo comprensibile e usabile. Quel che più conta è presentare un modello concettuale capace di guidare l'utente quando le cose non vanno come dovrebbero.

Un buon modello concettuale è la chiave per avere prodotti comprensibili, di facile uso e gradevole. La buona comunicazione è la chiave per ottenere buoni modelli concettuali.

Capitolo 6

Constraints, Discoverability e Feedback

In che modo si riesce a capire una cosa che non abbiamo mai visto prima?

Non c'è altro da fare che combinare l'informazione presente nel mondo esterno con quella che abbiamo in testa.

L'insieme di conoscenze che troviamo nel mondo comprende le affordance, i significanti visibili, le corrispondenze fra quelle parti degli oggetti che sembrano comandi o punti da manipolare, le azioni risultanti e i vincoli fisici, che limitano ciò che è possibile fare.

La conoscenza che abbiamo in mente comprende i modelli concettuali, i vincoli culturali, semantici e logici del comportamento, le analogie fra la situazione attuale ed esperienze precedenti.

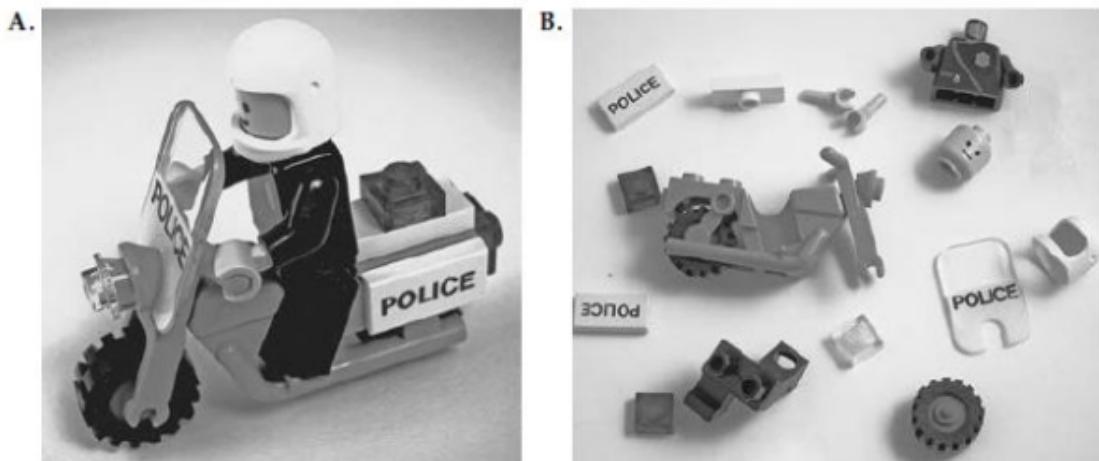


Figura 6.1: Un modellino lego.

Prendiamo come esempio il modellino Lego presente in figura: ha 15 pezzi, solo alcuni specializzati, molti altri sono di grandezza e forma uguale ma di colori diversi. Ma combinando i vincoli fisici con quelli culturali, semantici e logici si riesce a costruire il modellino senza istruzioni, mettendo ogni pezzo nella sua giusta posizione.

Vincoli fisici limitano le parti che possono andare insieme, i vincoli culturali e semantici impongono restrizioni precise a tutti i pezzi restanti e se rimane fuori qualche pezzo l'incastro è dettato dalla logica.

I vincoli sono indizi potenti, che limitano l'insieme delle azioni possibili. L'uso intelligente dei vincoli in sede di design permette alle persone di decidere prontamente il giusto corso d'azione, anche in una situazione del tutto nuova.

Possiamo categorizzare i vincoli in **quattro** classi:

- **Vincoli fisici:** si affidano a proprietà del mondo fisico, senza alcun bisogno di istruzioni o di addestramento. Nell'esempio della motocicletta Lego ritroviamo questo vincolo nei pezzi che si incastrano solo in un determinato verso.
- **Vincoli culturali:** si affidano alle abitudini culturali, sociali, comportamentali che possono cambiare nel tempo. Con vincolo culturale si intendono anche le convenzioni. Nell'esempio della motocicletta Lego ritroviamo questo vincolo nel saper determinare la collocazione delle luci: bianco all'anteriore e rosso al posteriore.
- **Vincoli semantici:** si affidano al significato della situazione per circoscrivere l'insieme delle azioni possibili, si basano sulla conoscenza della situazione e del mondo. Nel caso della motocicletta, c'è un'unica collocazione sensata per il motociclista, deve stare seduto guardando in avanti.
- **Vincoli logici:** dettati dalla semplice e pura logica umana. Se avanzasse un un solo pezzo per assemblare la motocicletta, grazie alla logica sapremo dove collocarlo nella sua giusta posizione.

Un buon designer può sfruttare questi vincoli per veicolare l'utente verso un modello mentale del prodotto che si avvicini il più possibile al modello concettuale desiderato ed in tal modo garantirgli una UX gradevole.

6.1 Vincoli e mapping

Vincoli e mapping a volte si confondono tra di loro. Una serie di interruttori mappati in maniera opportuna danno un vincolo logico che permette all'utente di non sbagliare, si intuisce perfettamente cosa verrà azionato da quell'interruttore posto in quel determinato punto. **Mapping forti diventano quasi dei vincoli logici.** L'assenza di vincoli e mapping genera, come detto più volte, frustrazione poiché crea una interfaccia poco chiara e difficile da comprendere.



Figura 6.2: Un interruttore per le luci di una stanza che imita la piantina del locale.

6.2 Funzioni Obbliganti

Le funzioni obbliganti sono una forma di vincolo fisico: consistono di situazioni in cui le azioni sono vincolate in modo che un passaggio mancato impedisce di procedere al successivo.

Sono il caso estremo di vincoli per impedire un comportamento inappropriato.

Non tutte le situazioni permettono l'intervento di vincoli così forti, ma il principio generale si applica negli ambiti più diversi.

Esaminiamo tre di questi metodi per applicare funzioni obbliganti:

- **Interlock:** obbliga a eseguire le operazioni nella sequenza dovuta. Usati soprattutto nell'ambito della sicurezza. **Per compiere un task si deve eseguire una serie di passi.**
- **Lock-in:** mantiene attiva una funzione impedendo che qualcuno la interrompa prematuramente. Usato molto in ambito informatico (e.g. ogni tentativo di uscita da un'applicazione senza salvare è prevenuto da un messaggio di allerta che chiede la conferma dell'intenzione). **Per finire un task si deve compiere un'azione.**
- **Lockout:** impedisce l'ingresso in uno spazio pericoloso o impedisce che succeda qualcosa. Può essere considerato l'opposto del lock-in. Un esempio di stampo informatico, sono gli alert "VM 18" dove dobbiamo dichiarare la maggiore età che si possono trovare su alcuni siti. **Per iniziare un task si deve compiere un'azione.**

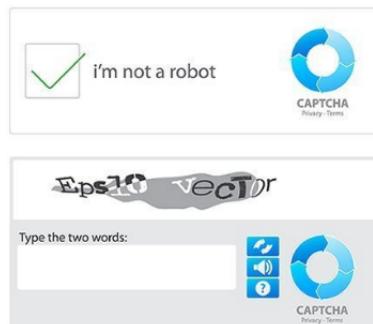


Figura 6.3: I captcha sono un esempio di interlocks.

6.3 Activity-Centered Control

Il mapping spaziale dei comandi non sempre è il più opportuno.

In molti casi è meglio avere interruttori diversi per attività diverse: **comandi centrati sulle attività.**

Quindi azionando un semplice comando si imposta una serie di oggetti per svolgere una determinata attività, senza comandarne uno per uno. In molti auditorium ci sono interruttori con indicazioni "video", "computer", "piena luce", "lezione" che impostano il microfono, le luci della sala, il proiettore e quant'altro, nel miglior modo per svolgere l'attività selezionata.

Questo schema è eccellente in teoria, ma nella pratica è difficile da realizzare bene, soprattutto è necessario valutare gli imprevisti e le possibili risoluzioni.

Il metodo è giusto, purché la gamma di attività sia scelta in modo da rispondere alle situazioni reali. Ma anche in quel caso saranno pur necessari dei comandi manuali, perché si presenteranno sempre esigenze inattese, che richiederanno una regolazione particolare dei dispositivi.

Capitolo 7

How do people do things

È facile imparare alcune azioni elementari per far funzionare un dispositivo tecnico. Ma cosa succede se le cose non vanno come dovrebbero? Come può l'utente accorgersene, e scoprire cosa fare?

Per chiarire meglio tutto questo è bene soffermarsi sulla psicologia umana e su un semplice modello concettuale dei modi in cui si sceglie e si valutano le nostre azioni. Da qui si passerà a esaminare il ruolo della cognizione e delle emozioni: piacere quando le cose funzionano senza intoppi, frustrazione quando le nostre aspettative iniziali sono bloccate.

7.1 I Golfi dell'Esecuzione e della Valutazione

Quando usiamo un oggetto, ci troviamo davanti due golfi: il **golfo dell'esecuzione**, nel quale cerchiamo di indovinare come funziona e cosa fare, e il **golfo della valutazione**, in cui si tratta di capire cosa succede. Il compito del progettista è aiutare le persone a superare i due golfi e renderli il meno profondi possibili.

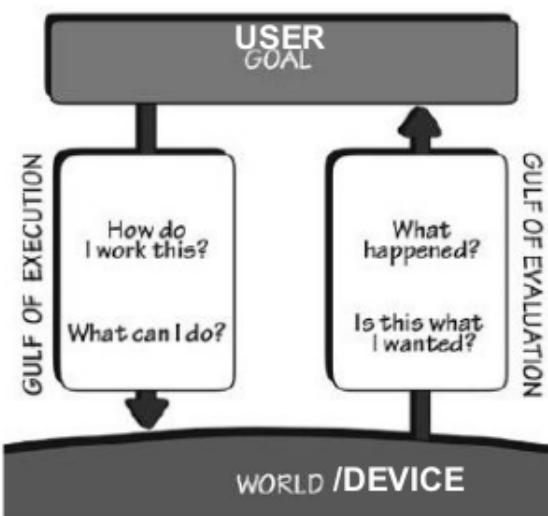


Figura 7.1: Golfo dell'esecuzione e golfo della valutazione.

Il **golfo della valutazione** corrisponde allo sforzo necessario per interpretare lo stato fisico del dispositivo e capire fino a che punto sono realizzate aspettative e intenzioni. Il Golfo è stretto quando il dispositivo fornisce informazioni sul proprio stato, in una forma facile da cogliere e interpretare, e corrispondente all'idea che abbiamo del suo funzionamento.

Quali sono gli elementi progettuali più importanti per superare il golfo della valutazione?

Il feedback e un modello concettuale adeguato.

Quali sono gli elementi progettuali più importanti per superare il golfo dell'esecuzione?

Significanti, constraints, mapping e un modello concettuale.

Entrambi i golfi sono presenti in molti apparati. Si incontrano spesso difficoltà, ma ogni volta vengono liquidate accusando se stessi. Di fronte a queste cose che ci si aspetterebbe di saper usare, si conclude semplicemente con *"sono stupido"*. Oppure, con dispositivi dall'aspetto più complicato, semplicemente ci si arrende, pensando di essere incapaci di utilizzarli. Queste spiegazioni sono entrambe sbagliate. **Le difficoltà hanno origine nel desing, non nell'utente.**

7.2 I sette stadi dell'azione

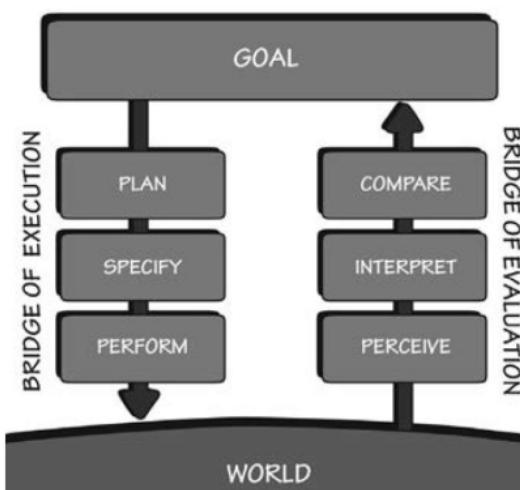
Un'azione implica due fasi: **eseguirla e valutarne gli effetti, fare e interpretare**. Sia l'esecuzione che la valutazione richiedono che si capisca come funziona una cosa e quali risultati produce. Entrambe le fasi influiscono sul nostro stato emotivo.

Le azioni specifiche fanno da ponte fra ciò che vorremmo veder realizzato e tutte le possibili azioni fisiche per arrivarci. Una volta specificato quali azioni compiere, dobbiamo effettuarle concretamente: questo è lo **stadio dell'esecuzione**. Dallo scopo discendono i tre stadi dell'esecuzione: **pianificare, specificare ed eseguire**.

La valutazione si articola anch'essa in tre stadi: **percepire, interpretazione, confrontare**.

Ecco così che abbiamo i **sette stadi dell'azione**: uno per lo scopo, tre per l'esecuzione e tre per la valutazione.

- **Scopo:** definire l'obiettivo.
- **Progettare:** l'azione da eseguire.
- **Specificare:** una sequenza d'azione.
- **Eseguire:** la sequenza specificata.
- **Percepire:** lo stato del mondo.
- **Interpretare:** la percezione.
- **Confrontare:** il risultato con lo scopo.



La maggior parte delle azioni non richiede tutti i sette stadi in sequenza, ma quasi nessuna attività si risolve in un’azione singola.

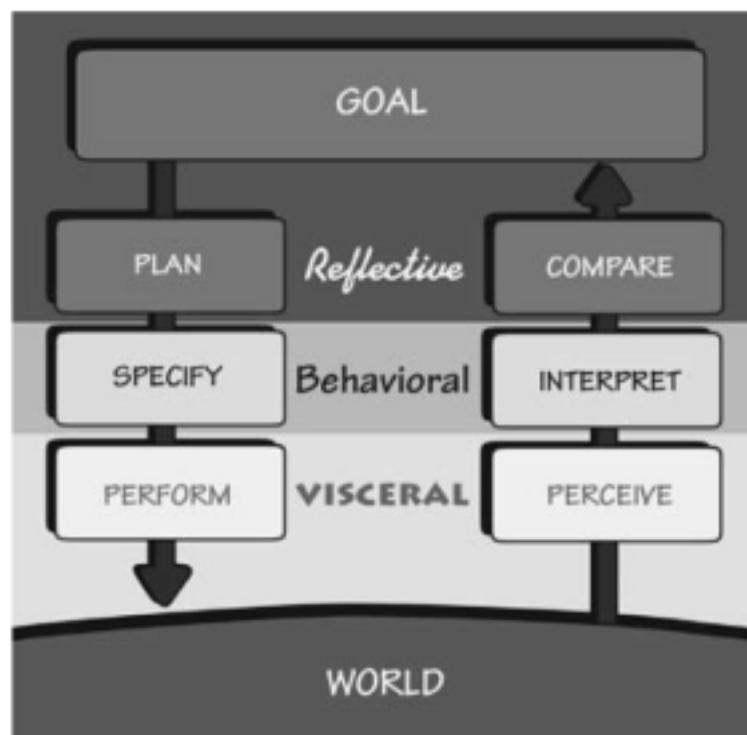
Di solito intervengono numerose sequenze e l’intera attività può durare ore o giorni. Ci sono molteplici circuiti di feedback, con cui i risultati di un’attività sono usati per indirizzare l’utente verso altre, in cui uno scopo genera scopi accessori, un progetto sotto-progetti. Ci sono attività in cui lo scopo originario è dimenticato, scartato o riformulato.

I sette stadi offrono uno schema per sviluppare nuovi prodotti o servizi. I golfi dell’esecuzione e della valutazione sono i punti più ovvi da cui partire, poiché entrambi offrono l’occasione di migliorare il prodotto. Tutto sta nello sviluppare capacità di osservazione per scoprirli.

7.3 Tre livelli di Processing

Gli stadi dell’azione possono essere associati con tre livelli di processing mentale: viscerale, comportamentale e riflessivo.

- **Livello viscerale:** è il livello più elementare che ci permette di rispondere prontamente in maniera subconscia, senza consapevolezza o controllo cosciente.
- **Livello comportamentale:** è la sede delle abilità apprese, attivate da situazioni che corrispondono al modello pertinente. In sede di design, il livello comportamentale è guidato dalle aspettative durante l’esecuzione e guidato dalle emozioni durante l’attesa di conferme di tali aspettative. Decide in che modo si compie un determinato task e in che modo si interpreta un determinato feedback.
- **Livello riflessivo:** è quello della cognizione conscia, è quindi a questo livello che si sviluppa la comprensione profonda e hanno luogo il ragionamento e i processi decisionali. Qui fanno capo i livelli più alti di emotività: è qui che avviene la soddisfazione e l’orgoglio, ma anche la frustrazione e il senso di colpa.



Per il progettista il livello riflessivo è forse il più importante dei tre. La riflessione è conscia e le emozioni che si producono a questo livello sono le più durature: quelle che attribuiscono responsabilità agli agenti casuali, come colpa, vergogna e orgoglio.

Le risposte riflessive sono anche parte integrante del ricordo degli eventi, che dura assai più a lungo dell'esperienza immediata o del periodo d'uso, che sono invece ambiti del livello viscerale e comportamentale.

È la riflessione che ci induce a consigliare un prodotto, a raccomandare l'uso o magari a sconsigliarlo.

I tre livelli di elaborazione contribuiscono tutti insieme a determinare il nostro stato emotivo e cognitivo. Funzioni riflessive di alto livello possono mettere in moto le emozioni più elementari, così come queste possono stimolare attività cognitive di tipo riflessivo.

7.4 I sette Principi Fondamentali della Progettazione

Il modello in sette stadi del ciclo d'azione può essere un prezioso sussidio per il design, in quanto suggerisce una lista di domande fondamentali. In generale, ogni stadio dell'azione richiede specifiche strategie progettuali, e, viceversa, presenta occasioni tutte sue di disastro.

Derivano quindi sette domande, a cui dovrebbe poter rispondere chiunque stia usando un determinato prodotto.

- Cosa voglio ottenere?
- Quali sono le sequenze d'azione alternative?
- Quale azione posso fare ora?
- Come faccio questa azione?
- Cosa è successo?
- Cosa significa?
- Va bene? Ho realizzato il mio scopo?



Il progettista ha la responsabilità di garantire che a ogni stadio dell'azione il prodotto fornisca l'informazione necessaria per la risposta.

L'informazione che serve a rispondere alle domande sull'esecuzione è il **feedforward**.

L'informazione che aiuta a capire quello che è successo è il **feedback**.

Il **feedforward** si realizza mediante l'uso opportuno di significanti, vincoli e mapping, anche il modello concettuale ha un ruolo importante. Il **feedback** è dato dall'informazione esplicita circa l'impatto dell'azione eseguita e anche qui una parte importante è svolta dal modello concettuale.

Sia il **feedback**, che il **feedforward** devono presentarsi in una forma facilmente interpretabile da chi utilizza il sistema. La presentazione deve corrispondere alla visione che le persone hanno dello scopo che vogliono realizzare e alle loro aspettative. L'informazione deve essere congruente con le esigenze umane.

Dalle risposte relative ai sette stadi dell'azione si ricavano sette principi fondamentali del design:

- **Visibilità:** è possibile scoprire immediatamente quali azioni sono possibili e qual è lo stato attuale del dispositivo.
- **Feedback:** c'è un'informazione completa e continua riguardo ai risultati delle azioni e allo stato attuale del prodotto o del servizio. Dopo aver eseguito un'azione, è facile determinare il nuovo stato.
- **Modello Concettuale:** il design fornisce tutta l'informazione necessaria per creare un buon modello concettuale del sistema, che favorisca la comprensione e la sensazione di controllo. Il modello concettuale potenzia sia la visibilità, sia la valutazione dei risultati.
- **Affordance:** affordance corrette sono fatte apposta per rendere possibili le azioni desiderate.
- **Significant:** un uso efficace dei significanti assicura la visibilità e un feedback efficiente e intellegibile.
- **Mapping:** la relazione fra i comandi e le rispettive azioni obbedisce ai principi del buon mapping, sostenuto, per quanto possibile, dalla disposizione spaziale e dalla contiguità temporale.
- **Vincoli:** fornire vincoli fisici, logici, semantici e culturali guidando l'azione e facilitandone l'interpretazione.

Questi sette principi sono mappati uno a uno sugli stadi d'azione dell'utente.

È bene concludere con una nota la parte dedicata a strumenti, metodi ed elementi per il design dello human-computer interaction: per molte attività quotidiane , gli obiettivi e le interazioni non sono ben definiti, **sono più di tipo opportunistico che pianificato**.

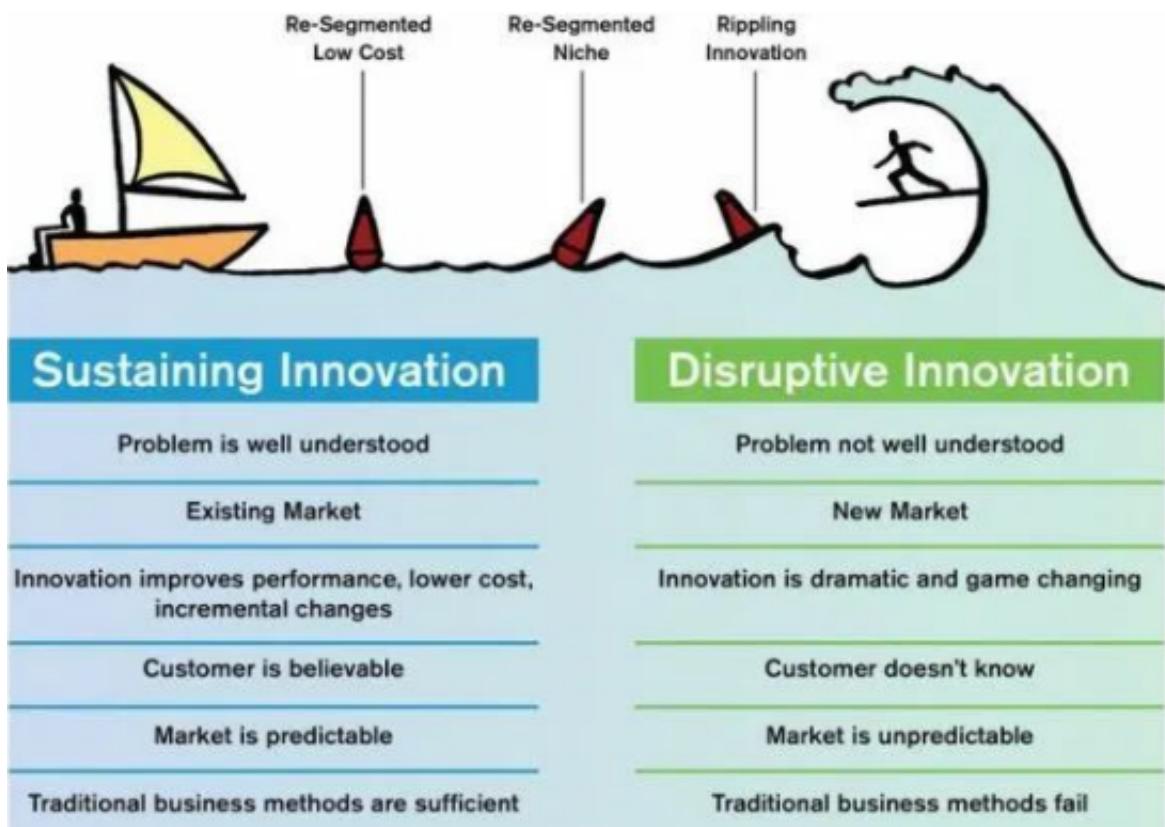
Le azioni opportunistiche sono quelle in cui il comportamento scaturito dalle circostanze prevale sulla pianificazione. Gli utenti in questi casi agiscono in maniera non controllata e quindi non prevedibile.

È difficile fare buon design per queste situazioni, anche attenendosi a tutti i principi esposti fino ad ora: l'utente che agisce in maniera opportunistica romperà questi schemi.

7.5 Disruptive Innovation

"People don't want to buy a quarter-inch drill. They want a quarter-inch hole!"

La maggior parte dell'innovazione è fatta come miglioramento incrementale di prodotti già esistenti. La **disruptive innovation** riguarda l'innovazione non lineare, consiste in un completo salto di binario. Vuole introdurre idee radicali che portano a progettare nuove categorie di prodotto nel mercato.



Ciò è effettuato riconsiderando l'obiettivo tramite un procedimento chiamato **root cause analysis**, che può essere decomposto in quattro fasi.

La **root cause analysis** è quindi fatta:

1. Descrivendo chiaramente il problema.
2. Tracciando il flusso degli eventi che ha fatto in modo che da una situazione di normalità emergesse un problema.
3. Distinguendo le cause principali dalle cause secondarie.
4. Tracciando un grafico che descrive le relazioni tra le cause principali e il problema.