Appunti Interazione Uomo Macchina

Design

Per design si intende sia il processo di progettazione e pianificazione, sia il risultato di questo processo. L'obiettivo principale del design di prodotto non è necessariamente quello di trovare una soluzione al problema specifico ma è piuttosto quella di comprendere il problema stesso nel suo insieme. Nel mondo del design il primo passo è sempre quello di capire perché il problema esiste e solo dopo aver appurato che l'origine di un problema non può essere eliminata o mitigata ci si adopera per cercare di risolverlo nello specifico. Nel design di prodotto ci si trova infatti spesso costretti a modificare i requirement e le specifiche di prodotto per andare in contro alle esigenze degli utenti e sacrificando funzionalità tecniche e qualità dell'implementazione software. Trovare il corretto bilanciamento fra esperienza utente, funzionalità e qualità tecnica è la parte più complessa dell'intero processo di sviluppo prodotto.

Il mondo del design è diventato talmente ampio che il termine design da solo ormai non ha quasi più significato. Esistono varie sotto discipline del design e con queste, numerose professioni, metodi di lavoro, scuole di pensiero e altrettante immancabili faide e lotte fra fazioni. Tra le varie sotto discipline del design troviamo l'interaction design che è l'attività di progettazione dell'interazione che avviene tra esseri umani e oggetti in generale. L'obiettivo principale dell'interaction design è quello di rendere macchine, servizi e sistemi usabili dagli utenti per cui sono stati pensati e realizzati e non solamente dai propri creatori. Le forze trainanti lo sviluppo di un prodotto dovrebbero esserre quindi gli utenti reali e i loro bisogni e non solo le tecnologie. Ciò che ci interessa quindi è il campo dell'interaizone uomo-macchina e uomo-computer cioè Human-Machine Interaction (HMI) e Human-Computer Interaction (HCI). L'interaction design ha a sua volta altre sottodiscipline che spesso si intrecciano tra di loro nei vari settori:

- Product Design : è un processo strategico di risoluzione dei problemi che guida l'innovazione eporta a una migliore qualità della vita attraverso prodotti, sistemi, servizi e esperienze innovative. Nel design di prodotto si progettano beni e servizi il cui obiettivo principale è quello di essere utilizzati da quanti più utenti possibili migliorandone la vita
- User Experience (UX) design : è il processo volto ad aumentare la soddisfazione e la fedeltà degli utenti migliorando l'usabilità, la facilità d'uso e il piacere fornito nell'interazione tra il cliente e il prodotto. L'UX designer ha il compito di far vivere all'utente la miglior esperienza possibile evitando che l'oggetto induca sensazioni di frustrazione e delusione.
- User Interface (UI) design : ha come focus il 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. Dallo studio dell'UX si crea uno schema di interazione che poi viene passato allo UI designer che si occupa di creare l'interfaccia grafica che l'utente vedrà e con cui interagirà. UI designer non costruisce l'interfaccia utente ma progetta l'aspetto estetico e la struttura dell'interfaccia così che questa durante l'utilizzo induca l'utente a seguire l'esperienza che è stata progettata per lui. L'UI designer produce un wireframe per implementare la reale interfaccia del prodotto o servizio.

Il processo di Product Design è un processo strutturato che include varie figure e discipline. L'interfaccia vera e propria viene implementata solo alla fine del percorso di progettazione da figure con profilo tecnico informatico che comprendo le richieste provenienti dalle fasi precedenti e le implementano in un prodotto finito.

Human Centered Design

Lo Human Centered Design (HCD) è una metodologia di progettazione che parte dai bisogni, dalle capapcità e dai comportamenti umani, adattando la progettazioni a quei bisogno, quelle capacità e dai comportamenti umani. Lo HCD è un approccio che mette al centro dell'attenzione l'utente e le sue esigenze a differenza di un approccio tecno-centrico che non si concentra a cercare tecnologie che possano risolvere i problemi degli utenti ma cerca problemi da risolvere con nuove tecnologie. Per questo lo HCD parte dall'osservazione dell'utente e dai suoi comportamente e solo identifica la tecnologia necessaria. 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 ed è qui che i progettisti devono concentrare la loro attenzione non a non far presentare errori agli utenti, poiché inevitabile, ma progettare un interfaccia che guidi l'utente nella risoluzione in modo che l'utente non si senta frustrato. L'obiettivo della forma di pensiero HCD deve dunque essere quello di creare nell'utente empatia verso il sistema.

Possiamo schematizzare un processo HCD come un fllusso continuo ed iterativo che attraversa le seguenti fasi:

- Specificare il contesto d'uso : identificare gli utenti che utilizzeranno il prodotto, per cosa lo utilizzeranno e sotto quale condizioni e vincoli
- Specificare i requisiti : identificare i requisiti e gli obiettivi dell'utente che devono essere raggiunti dal prodotto
- Progettare la soluzione : realizzare un prodotto finale attraversando varie fasi di bozze e prototipi
- Testare e valutare : testare il prodotto con gli utenti e valutarne l'efficacia e l'efficienza

In tutto questo è necessario anche considerare l'usabilità cioè il grado in cui un prodotto può essere utilizzato da utenti specifici per raggiungere obiettivi specifici con efficacia, efficienza e soddisfazione in un contesto d'uso specifico ed è imprescendibile.

Progettazione delle interfacce

Il problema è che il buon design non è universale, poiché l'esperienza di interazione è soggettiva e quindi dipende più dalla persona che dal prodotto, ma esistono due proprietà fondamentali che qualsiasi progetto deve avere:

- Discoverability (rilevabilità) : la capacità di un sistema di veicolare e comunicare i propri usi all'utente a prima vista. Per avere una buona discoverability sollitamente si usa la visibilità, cioé se le funzioni dell'oggetto sono visivamente eclatanti è probabile che abbia una buona discoverability
- Understanding (comprensibilità) : la capacità del prodotto di farsi usare correttamente dall'utente cioé la misura di quanto bene l'utente capisce come usare le funzioni del prodotto.

È fondamentale comprendere il concetto di "Design of Useful Things", il quale si basa sulla premessa che quando le cose funzionano correttamente, vengono dimenticate immediatamente, mentre quando presentano problemi, non vengono mai dimenticate. Questo fenomeno è noto a tutti ed ho dovuto perché ormai viviamo in una società in cui le cose devono andare bene per definizione, mentre se qualcosa va storto scattano una serie di reazioni che portano la persona a provare senzazioni e emozioni spiacevoli. Il design deve quindi preoccuparsi di come funzionano le cose, come vengono controllate e dalla natura delle interazione che questi oggetti e sistemi abilitano con gli utenti. Questo vuol dire che una cattiva esperienza utente induce l'utilizzatore ad abbandonare l'utilizzo del prodotto e etichettare quel prodotto e azienda come negativa e perciò bisogna fare di tutto per evitarlo. Il problema è che le macchine sono concepite, progettate e costruite per funzionare perfettamente ma secondo regole di comportamento rigide a differenza dell'uomo che è aleatorio, versatile, variabile, volubile e intuitivo e data questa differenza bisogna evitare che le macchine obblighino le persone a smettere di comportarsi da umani inducendole a comportarsi da macchine. Quindi quando la macchina fa la cosa sbagliata, tipicamente l'utente si senta incapace e perciò deve essere ribaltata la prospettiva in modo che quando le cose vanno male la colpa non è mai dell'utente ma è sempre della macchina e quindi del progettista, che deve progettare per come le persone sono e non per come vorrebbe che fossero.

L'incidente di three mile island è un eclatante esempio pratico di mal progettazione delle interfacce. Il 28 marzo del 1979 nella contea di Daulphin, in Pennsylvania ci fu il più grave incidente nucleare avvenuto negli Stati Uniti che ha portato al rilascio di piccole quantità di gas radioattivi e di iodio radioattivo nell'ambiente. Il reattore era ad un regime di potenza del 97% e il sistema di raffreddamento secondo si era spento portando un considerevole aumento di pressione al circuito primario causando l'apertura di una valvola PORV di rilascio posta sul pressurizzatore e portò allo SCRAM, l'arresto di emergenza mediante l'inserimento delle barre di controllo. A questo punto la valvola di rilascio non si richiuse e gli operatori non si resero conto del problema, poiché non vi era nella strumentazione l'indicazione della reale posizione della valvola. La strumentazione era infatti legata all'alimentazione del motore della valvola e non alla reale posizione della valvola. Fu così che il circuito di raffreddamento primario si vuotò parzialmente e il calore residuo del nocciolo del reattore non potè essere smaltito. Gli operatori non poterono

diagnosticare correttamente cosa avveniva e reagire in maniera adeguata. La strumentazione carente della sala controllo e l'addestramento inandeguato risultarono essere le cause principali dell'incidente.

Principi fondamentali dell'interazione

Un buon design produce un'esperienza piacevole, e ciò è necessario perché grazie all'esperienza si crea un bel ricordo mentre quando la tecnologia si comporta in maniera inaspettata si provano emozioni negative. Un utente che comprende il corretto funzionamento di un prodotto e si sente padrone sarà soddisfatto e orgoglioso. Cognizione ed emozione sono profondamente legate. La visibilità o discoverability di un prodotto è fondamentale per suscitare queste emozioni ed è il risultato di 5 concetti psicologici fondamentali: affordance, singnificanti, mapping, feedback e vincoli. C'è poi un sesto principio che come vedremo non è legato a degli elementi specifici dell'interfaccia ma piuttosto al concetto generale su cui si va a basare il funzionamento del sistema cioè il modello concettuale del sistema

Il termine affordance, che vuol dire invito, indica la relazione tra le proprietà di un oggetto e le capacità dell'agente che determina come l'oggetto possa essere utilizzato. Le affordance non sono quindi proprietà oggettive di un sistema o prodotto ma piuttosto relazioni prodotto-utente che se stabilite abilitano o disabilitano specifiche modalità di interazione fra le parti, ad esempio una sedia sembra fatta apposta per sostenere qualcosa e invita alla seduta. Oltre all'affordance esiste l'anti-affordance che invece nega alcuni modi di interagire con l'oggetto, ad esempio degli spunzoni negano che un utente ci si sieda sopra. Le affordance e le anti-affordance per abilitare o disabilitare una particolare modalità di interazione fra oggetto ed utente devono essere percepibili, discoverable.

Per dare visibilità ad un'affordance si usano i significanti, che sono un modo per indicare dove effettuare un'azione. Molto spesso i significanti sono indispensabili poiché la maggior parte delle affordance sono invisibili. Possono essere:

- Voluti o intenzionali : come un'etichetta o un'icona
- Accidentali o non intenzionali : come un sentiero tracciato oppure delle persone in fila

Nel design i significanti sono molto più importanti delle affordances, perché comunicano come usare il prodotto o l'interfaccia e per associare un affordance ad un azione, nella maggior parte dei casi, si usano le convenzioni culturali.

Il 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 nella programmazione di interfacce, in particolare nel posizionamento dei significanti. La disposizione di significanti può comunicare sull'interfaccia e le sue funzionalità, soprattutto se si sfrutta una corrispondenza spaziale fra la collocazione dei comandi e quella dei dispositivi comandati. Il modo migliore per fare mapping è quello naturale con forme e colori in modo da facilitare le associazioni nel nostro cervello.

Il feedback è la comunicazione del risultato di un'azione che l'interfaccia da all'utente come risposta. È necessario che sia immediato poiché un ritardo potrebbe far rinunciare l'utente a svolgere l'attività e deve essere informativo, non portando con sè troppe informazioni ma deve far capire che è stato prodotto il risultato che ci si aspetta. Una caratteristica importante è la semplicità e essenziale, evitando troppi annunci che non siano veramente importanti perché potrebbe portare l'utente a non leggerli.

Un modello concettuale è una descrizione altamente semplificata delle funzionalità di un sistema che non punta a essere completa o accurata ma utile. I modelli semplificati sono preziosi e utili fintanto le ipotesi che li supportano sono vere. Il modello concettuale esprime come il designer vuole che l'utente percepisca il prodotto e serve per poi progettare un'interfaccia in modo che l'utente possa capire come funziona il sistema attraverso affordances, significanti e mapping. Un modello mentale è un modello concettuale nella mente dell'utente che rappresenta il modo in cui secondo lui funzionano le cose, ed è soggettivo e diverso da utente ad utente. Più è grande la differenza tra modello mentale e quello concettuale e 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, che gli venga trasmesso da terzi.

L'insieme di tutta l'informazione accessibile, come l'aspetto dell'apparecchio, cosa hanno imparato da oggetti simili in passato, il sito web e il libretto di istruzione, costruiscono l'immagine di sistema. 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 ed il modello concettuale dell'utente deriva dall'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 le informazioni adeguate per rendere il prodotto comprensibile ed usabile.

Vincoli

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

- Fisici : si affidano alle proprietà del mondo fisico, senza alcun bisogno di istruzioni o di addestramento
- Culturali : si affidano alle abitudini culturali, sociali, comportamentali che possono cambiare nel tempo
- Semantici : si affidano al significato della situazione per circoscrivere l'insieme delle azioni possibili, si basano sulla conoscenza della situazione e del mondo
- Logici : dettati dalla semplice e pura logica umana.

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 e in tal modo garantirgli un UX gradevole. Mapping forti possono diventare dei vincoli logici, ed entrambi sono necessari per rendere l'interfaccia chiara e facile da comprendere.

Le funzioni obbliganti sono una forma di vincolo fisico e consistono in situazioni in cui le azioni sono vincolate in modo che un passaggio mancato impedisca di procedere al successivo. Sono il caso estremo di vincoli atti ad impedire un comportamento inappropriato. Tre di questi metodi sono:

- interlock : obbliga ad eseguire una serie di operazione nella sequenza corretta prima di avviare l'azione richiesta. Sono usati soprattutto come sistemi di sicurezza nei macchinari industriali ma anche nel mondo software come per esempio nel caso dei sistemi Captcha
- lock-in : mantiene attiva una funzione impedendo che qualcuno la interrompa prematuramente. Per finire un task si deve compiere un'azione
- lock-out : impedisce l'ingresso in uno spazio pericoloso o impedisce che succeda qualcosa, può essere considerato l'opposto del lock-in. Per accedere ad un task si deve compiere un'azione

Il mapping spaziale dei comandi non è sempre il più opportuno, in molti casi è meglio avere interruttori diversi per attività diverse : comandi centrati sulle attività. Con i comandi centrati sulle attività possiamo usare un semplice comando per impostare una serie di oggetti per svolgere una determinata attività, senza comandarli uno ad uno. Questo metodo funziona particolarmente bene con utenti esperti poichè nella pratica è difficile da realizzare e soprattutto è necessario valutare gli imprevisti e le possibili soluzioni.

Come le persone fanno le cose

Per capire come gli utenti si approcciano ad un prodotto e come imparano ad usarlo, è importante soffermarsi anche su come la mente umana lavora e come vengono scelte le azioni da compiere, che porteranno sia il piacere quando le cose funzionano senza intoppi e sia quando si frustrano perché le cose non funzionano come dovrebbero.

Quando usiamo un oggetto, ci si trova davanti due golfi:

- Il golfo dell'esecuzione nel quale si cerca di indovinare cosa fare
- Il golfo della valutazione nel quale si cerca di capire cosa è successo

Il compito del progettista è quello di aiutare gli utenti a superare i due golfi e renderli meno profondi possibili.

Il golfo dell'esecuzione corrisponde allo sforzo necessario per determinare come operare sul dispositivo, quali azioni sono possibili e come eseguirle. Il golfo è stretto quando il dispositivo fornisce un buon mapping tra le azioni e gli effetti, quando le azioni sono visibili e quando il dispositivo fornisce un buon feedback. Gli elementi progettuali più importanti per superare il golfo dell'esecuzione sono il mapping, la visibilità, i vincoli e un modello concettuale adeguato.

Il golfo della valutazione corrisponde allo sforzo necessario per interpretare lo stato fisico del dispositivo e capire fino a che punto sono state realizzate le aspettative e le intenzioni iniziali. Il golfo è stretto quando il dispositivo fornisce informazioni sul proprio stato, in una forma facile da cogliere e interpretare. Gli elementi progettuali più importanti per superare il golfo della valutazione sono il feedback e un modello concettuale adeguato.

Entrambi i golfi sono presenti in molti apparati ma ed è importante tenere a mente che l'utente non si deve sentire stupido quando utilizza un prodotto e perciò le difficoltà hanno origine nel design e non nell'utente.

Compiere un'azione implica due fasi:

- esecuzione e valutazione degli effetti
- fare e interpretare

Sia l'esecuzione che la valutazione richiedono che si capisca come funziona l'oggetto su cui si applica l'azione e quali risultati essa produce. Entrambe le fasi influiscono sullo stato emotivo dell'utente. Identificato il goal, o scopo, l'utente discende attraverso i tre stadi dell'esecuzione:

- pianificare : definire l'obiettivo
- specificare : costruire una sequenza d'azione
- eseguire : eseguire la sequenza d'azione

La valutazione si articola anch'essa in tre stadi:

- percepire : percepire lo stato del mondo
- interpretazione : elaborare la percezione
- confrontare : rapportare il risultato allo scopo

Ecco così che si hanno i 7 stadi dell'azione: unp per lo scopo, tre per l'esecuzione e tre per la valutazione. La maggior parte delle azioni non richiede che si percorrano tutti e 7 stadi in sequenza, ma quasi nessuna attività si risolve tramite un'azione singola. 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, offrendo entrambi spunti per migliorare il prodotto.

Il modello a 7 stadi del ciclo d'azione è un prezioso sussidio per il design, in quanto introduce una lista di domande fondamentali. In generale, ogni stadio dell'azione richiede specifiche strategie progettuali, e, viceversa, presenta occasioni proprie di disastro.

Ne derivano dunque 7 domande, a cui dovrebbe poter rispondere chiunque:

- Cosa voglio ottenere?
- Quali sono le sequenze d'azione alternative?
- Quale azione posso fare ora?
- 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 neccessaria per proseguire correttamente. L'informazione che serve a rispondere alle domande nelle fasi attuative è definita come feedforward. L'informazione che aiuta a capire quello che è successo nelle fasi percettive è definita invece come 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'immediato cambiamento di stato che il prodotto deve mostrare all'utente e, anche qui, una parte importante è svolta dal modello concettuale. Sia il feedback sia 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 erogata deve essere immediatamente comprensibile. Dalle risposte relative ai sette stadi dell'azione si ricavano sette principi fondamentali del design:

- visibilità : è bene che sia facile scoprire immediatamente quali azioni sono possibili e qual è lo stato attuale del dispositivo
- feedback : è opportuno che ci sia un'informazione completa e continua riguardo risultati delle azioni e allo stato attuale del prodotto o del servizio. Dopo aver eseguito un'azione, deve essere facile determinare il risultato
- modello concettuale : il design dovrebbe fornire tutta l'informazione necessaria per creare un buon modello concettuale del sistema, che favorisca la comprensione e la sensazione di controllo da parte dell'utente. Il modello concettuale potenzia sia la visibilità, sia la valutazione dei risultati.
- affordance : è bene che le affordance siano fatte apposta per rendere possibili le azioni desiderate e impossibili quelle indesiderate
- significanti : un uso efficace dei significanti assicura la visibilità e la comprensibilità dei comandi
- mapping : è necessario che la relazione fra i comandi e le rispettive azioni obbedisca ai principi del buon mapping, sostenuto, per quanto possibile, dalla disposizione spaziale e dalla contiguità temporale
- vincoli : bisogna fornire vincoli fisici, logici, semantici e culturali, in modo tale da guidare l'azione e facilitandone l'interpretazione

Questi 7 principi sono mappati uno ad uno sugli stadi d'azione dell'utente. La maggior parte delle azioni svolte quotidianamente, goal e intenzioni non sono davvero specificati: sono opportunistici anziché pianificati. Le azioni opportunistiche sono quelle in cui il comportamento scaturisce delle circostanze. 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 in ogni caso questi schemi.

La mente umana è infatti molto complessa, non esiste un indicatore che separi pensiero subconsio (di cui non siamo consapevoli poiché nascosto e veloce) e pensiero conscio (di cui siamo consapevoli poiché elaborato e orientato all'analisi). Il pensiero conscio interviene per l'apprendimento ma, al termine della fase iniziale, pratica e studio producono quello che gli psicologi chiamano "overlearning", una fase finale in cui un'abilità è stata "overlearned" e i suoi gesti verranno svolti in maniera automatica e senza sforzi.

Il modo in cui ricordiamo le esperienze anche se le azioni sono fatte dal subconscio avviene attraverso la memoria esponenziale che ricorda bene gli elementi importanti mentre quelli meno importanti li cerca di ricordare attraverso l'esperienza.

Distinguiamo 2 tipi di memoria:

- memoria dichiarativa : utilizzata per recuperare informazioni fattuali
- memoria procedurale : utilizzata per recuperare informazioni procedurali

Un altro elemento determinante per il passaggio da un pensiero subconsio a uno conscio è lo stato emozionale. La divisione in conscio e subconscio può essere ulteriormente affinata andando a dividere il modo in cui il cervello fa processing in tre livelli procedurali:

- livello viscerale : è il livello più elementare, permette di rispondere prontamente in maniera subconscia, senza consapevolezza o controllo cosciente
- livello comportamentale : è la sede delle abilità apprese durante circostanze più o meno simili a quelle attuali. Durante l'esecuzione, il livello comportamentale è guidato dalle aspettative, e durante l'attesa di conferma di tali aspettative è invece guidato dalle emozioni. Il livello comportamentale stabilisce in che modo si compie una determinata azione e in che modo si interpreta un determinato feedback
- livello riflessivo : è il livello della cognizione conscia, è qui che si sviluppa la comprensione profonda e hanno luogo il ragionamento e i processi decisionali. Qui fanno capo i livelli più alti di emotività: soddisfazione e orgoglio, ma anche frustrazione e senso di colpa

Veicolare informazioni all'utente mentre egli si trova nel livello riflessivo è estremamente efficace. Al livello riflessivo il suo pensiero è conscio e le emozioni che egli produce sono le più durature. Gli stimoli riflessivi sono parte integrante del ricordo degli eventi, è importante quindi creare nell'utente ricordi positivi mentre egli è in questo stadio dell'azione, perché tali ricordi sono i più duraturi. Inoltre è la riflessione, intesa come pensiero cosciente, che induce a consigliare un prodotto e raccomandarne l'uso o magari a sconsigliarlo

Errore Umano

La maggior parte degli incidenti industriali è causata da errore umano ma ciò non è causato dall'uomo che è incopetente ma bensì dalla mala progettazione e un cattivo design. Nei nuovi dispositivi sempre più spesso viene richiesta la massima vigilanza di tutte le attività o la memorizzazione di procedure complicate e l'interruzione è una della cause che più frequentemente portano all'errore umano.

Uno dei più grandi problemi è l'atteggiamento delle persone verso gli errori commessi che a causa delle perdite economiche o danni a persone, cercano sempre di trovare un responsabile e non una soluzione così che lo stesso errore continuerà a presentarsi. Per evitare di incorrere nuovamente nell'errore, quando esso viene commesso, è bene studiarne le cause e ridisegnare il prodotto.

La Root Cause Analysis consiste nell'indagare l'incidente finché non si trova la singola causa che ne è l'origine ovvero il momento nel tempo quando effettivamente qualcuno ha preso decisioni o eseguito azioni sbagliate e, una volta fatto ciò accertare da cosa è derivato lo sbaglio e non fermarsi solamente al colpevole. Cercare di trovare la causa di un incidente però ha due difetti:

- La maggior parte degli incidenti non ha una sola causa. Da qui il modello a groviera degli incidenti di James Reason
- Solitamente l'analisi delle cause profonde si ferma non appena trovato un errore umano

Se una macchina smette di funzionare per un guasto o un malfunzionamento si cerca di capire come mai si è rotta o cosa l'ha portata a guastarsi. È opportuno fare lo stesso quando si scopre un errore umano: individuarne le cause. Quando durante l'analisi delle cause profonde si incontra, nella concatenazione di cause ed effetti, un errore umano, il lavoro è appena cominciato: bisogna capire perché l'errore è accaduto e cosa si può fare per prevenirlo.

L'analisi delle cause profonde mira a determinare la causa prima di un evento,la vera causa di fondo, e non la causa immediata. Perciò la Toyota ideò il sistema di controllo dei suoi prodotto chiamato "Procedura dei 5 perché" che consiste nel chiedersi 5 volte il perché di un errore per arrivare alla causa principale e non soffermarsi alla prima causa apparente.

Quando le persone sbagliano, bisogna cambiare il sistema in modo da evitare l'errore e, se non è possibile eliminarlo del tutto, almeno fare in modo di ridurne gli effetti. Se il sistema lascia sbagliare gli utenti è mal progettato, se il sistema induce all'errore, allora è progettato malissimo.

Il fatto che le persone sbagliano succede perché il design si concentra sulle esigenze del sistema e delle macchine, non su quelle degli utenti.

Si definisce errore umano ogni deviazione del comportamento appropriato oppure ogni comportamento che si discosta da quello generalmente accettato come giusto o adeguato. Errore è il termine generale per tutte le situazioni sbagliate. È possibile dividere gli errori in due classi:

- Lapsus o slips : si ha quando s'intende eseguire un'azione e si finisce per eseguirne un'altra. Nel caso di lapsus, l'azione eseguita non è quella voluta. I lapsus si hanno nelle fasi attuative e percettive dell'azione. Ci sono due tipi principali di lapsus:
  - Di azione : si esegue un azione sbagliata.
  - Di memoria : si dimentica di eseuire un'azione o di valutarne i risultati.
- Mistakes : si ha quando si è sbagliato un goal o lo scopo. Da quel momento in poi le azioni, anche se esguite correttamente, fanno parte dell'errore essendo di per sé inappropriate. In questo tipo di errore l'azione è corretta ma l'intenzione no. I mistakes si hanno nele fasi di pianificazione e valutazione dell'azione. I mistakes si suddividono in:
  - Rule based : la diagnosi della situazione è giusta, ma poi viene scelto un corso d'azione sbagliato, seguendo una regola operativa errata
  - Knowledge based : la diagnosi della situazione è sbagliata
  - Memory lapse : un passaggio viene dimenticato nel momento in cui si fissano gli obiettivi o si esegue una procedura o se ne valutano i risultati

Per prevenire gli errori umani possiamo:

- Comprendere le cause dell'errore
- Controlli di sensibilità
- Rendere possibile l'annulamento delle azioni
- Rendere più semplice la scoperta e comprensione degli errori
- Aiutare l'utente a compiere correttamente le azioni

I novizi del sistema cadono in mistakes poiché non hanno una base di conoscenza adeguata, mentre gli utenti esperti che usano il sofware da molto tempo commettono errori di lapsus poiché tendono ad esguire i compiti in maniera automatica affidandosi al controllo del subconscio. I mistakes nascono da informazioni ambigue o poco chiare sulla stato del sistema e dalla mancanza di un buon modello concettuale. Le interruzioni sono una delle più grandi cause di errore, soprattutto i lapsus. Quando un'attività viene interrotta da qualche evento, il costo in attenzione è molto maggiore della perdita di tempo causata dell'interruzione. Per riprendere il lavoro è necessario ricordare precisamente il precedente stato dell'attività Alcune cause di errore umano sono anche i feedback usati in omdo sbagliato. Tanti segnali che occorrono i diverse situazioni in maniera ricorrente possono portare le persone a disconnettere i segnali o a non farci più caso ed a quel punto non ci si accorgerà più quando occorre un messaggio importante. I segnali quindi devono essere ponderati e usati solo quando necessario. Un segnale parlato è più efficace se usato in modo intelligente.

Per la prevenzione dell'errore utilizzare:

- Vincoli: aggiungendo vincoli alle azioni cosicché controlli confondibili vengono piazzati lontano l'uno dall'altro oppure usare moduli separati per controlli non collegati.
- Undo: comando che annulla le operazioni effettuate in precedenza.
- Messaggi di errore e di conferma : visualizzare un messaggio con sia l'azione da compiere che l'oggetto interessato con azione annulla o prosegui.
- Controlli di sensibilità : controlli che verificano se l'operazione richiesta sia sensibile o ragionevole.

Tutto ciò è riconducibile all'esempio della groviera degli incidenti di James Reason. Quindi per mitigare dell'errore usando metafora formaggio svizzero:

- Aumentare il numero di controlli (fette di formaggio)
- Diminuire la probabilità di errore (buchi nel formaggio)
- Allertare l'operatore umano quando ci si avvicina a un errore (buchi si sono allineati)

Le interfacce utente

Un interfaccia è qualcosa che sta fra due facce, un punto di contatto tra due sistemi che cercano di comunicare. Le interfacce possono far comunicare due macchine fra loro oppure possono far comunicare l'uomo con la macchina. Solitamente è sempre composta da due parti: uno strumento appartenente ad una persona che serve per compiere un'azione e l'intefaccia che è ciò serve per guidare l'utente nell'esecuzione dell'azione.

Quando parliamo di interfaccia utente intendiamo lo spazio di un sistema dove avviene l'interazione tra uomo e macchina e il loro obiettivo è quello di consentire all'utente di controllare e far funzionare il sistema in modo efficente. Quindi l'interfaccia deve essere progettata in modo da semplificare l'interazione fra l'uomo e la macchina per poi pensare a migliorare la User Experience. Una buona interfaccia massimizza la quantità di informazioni evitando comunque di sovraccaricare l'utente che deve riuscire ad utilizzare il sistema con il minimo sforzo fisico e cognitivo.

livelli di interfacce:

- Human Interface Device (HID)
- Human Machine Interface (HMI)
- Human Computer Interface (HCI)

Le interfacce utente tipicamente sono organizzate sulla base dei sensi che utilizzano per stabilire l'interazione tra uomo e macchina:

- Tactile UI
- Visual UI
- Auditory UI
- Olfactory UI
- Gustatory UI
- Equilibrial UI

La maggio parte delle interfacce però utilizzano più di un senso per stabilire il collegamento e vengono chiamate Composer User Interface (CUI). Tra le più note abbiamo:

- Graphical User Interface (GUI) : composta da interfaccie gradiche e tattile
- Multimodal User Interface (MUI) : composta da interfaccie che utilizzano più di un senso

Le CUI poi possono essere categorizzate in tre macrocategorie:

- Standard : usano dispositivi standard come tastiere, mouse e monitor
- Virtual : bloccano all'utente l'interazione con il mondo reale e creano un mondo virtuale che funge da interfaccia
- Augmented : non bloccano l'utente dalla percezione del mondo reale ma lo vanno ad arricchire, espandendola con un mix tra contenuti reali e virtuali

Le CUI possono essere categorizzate anche tramite il numero di sensi che utilizzano. Ad esempio, lo Smell-O-Vision è una CUI standard 3S (3 sensi) che utilizza la vista, l'olfatto e l'udito. Se si aggiungesse un quarto senso (come ad esempio le poltronne mobili del cinema) diventerebbe 4S. Quando un interfaccia utente interagisce con tutti i sensi umani viene chiamata Qualia Interface.

Human Interface Device (HID)

Un HID è un tipo di dispositivo informatica spesso usato da umani che consente l'interazione input/output tra umani e computer. Con HID indichiamo sia i device fisici, sia il protocollo USB-HID. Gli HID standard sono stati adottati la prima volta principalemente per semplificare il processo di installazione di ogni device. Tutti i dispositivi definiti HID invece inviano pacchetti auto-descrittivi che possono contenere qualsiasi tipo di dato e formato. Un driver HID su computer analizza i dati e consente l'associazione dinamica dell'I/O dei dati con le funzionalità dell'applicazione.

Nel protocollo HID ci sono 2 entità:

- Host : comunica con il device e riceve dati in input dal device in base alle azioni dell'umano e invece gli output attraversano il device e arrivano fino all'umano.
- Device : è un entità che interagisce direttamente con un umano

Questo protocollo rende l'implementazione di device molto semplice inviando all'host dei packetti di dati chiamati "HID descriptor" che descrivono come sono fatta i pacchetti del device e indicano:

- N° di pacchetti che il device supporta
- Dimensione dei pacchetti
- Lo scopo di ogni byte e bit nel pacchetto

Il device tipicamente memorizza HID descriptor in ROM così non ha bisogno di capire o analizzare l'HID descriptor. Ci si aspetta che l'host sia più complesso del device, ed ha bisogno, prima di comunicare con il device, di ricevere e analizzare l'HID descriptor. Dato che non tutti gli host potrebbero essere capaci di interpretare l'HID descriptor, HID descrive anche il boot protocol con pacchetti di formato predefinito

HID è stato esteso ad una serie di protocolli:

- Bluetooth HID
- Serial HID
- ZigBee input device
- HID over I²C
- HOGP

Le periferiche HID sono organizzate in 2 categorie:

- Di input : basati su sensori che convertonon la realtà fisica in segnali elettrici.
- Di output : basati su attuatori che convertono segnali elettrici in perturbazioni nel mondo reale.

Le varie classi di HID sono:

- Testi e caratteri
- Posizioni (sistemi di puntamento)
- Suoni
- Immagini
- Parametri ambientali
- Posizione
- Parametri fisiologici e biologici

HID testi e caratteri

Il primo dispositivo HID di testi e caratteri più comune è la tastiera ma sono disponibili più tipi in base ad ogni particolare necessità e variano a seconda della

dimensione, dal formato (ANSI o ISO) e del sistema operativo su cui verrano utilizzati. I layout da prendere in considerazione sono:

- Layout fisico : corrisponde al posizionamento dei tast sulla tastiera
- Layout visuale : corrisponde all'arrangiamento dei simboli che appaiano sui tasti
- Layout funzionale : corrisponde all'associazione tasto-significato all'interno di un software

Il layout design più utilizzato dalle popolazioni latine è il QWERTY. Esistono inoltre tastiere multifunzione che permettono di estendere le tastiere standard con altri tasti e mappature ma necessitano di driver aggiuntivi per funzionare.

Un altro dispositivo di HID testi e caratteri è il lettore di codice a barre poiché scannerizza è una serializzazione di caratteri e per lo stesso motivo lo è anche il QR code. I tag RFID sono un altro dispositivo di HID testi e caratteri che permettono di identificare un oggetto o una persona tramite un segnale radio che può essere passivo o attivo. Anche NFC è un dispositivo di HID testi e caratteri come i tag RFID ma funziona con distanza minore, è più lento ma la comunicazione è bidirezionale.

HID sistemi di puntamento

Sono dispositivi che trasferiscono input di tipo spaziale verso un computer come movimenti fisici dell'utente attraverso movimenti di punatori o altri cambiamenti visivi. Si definisce in merito la legge di Fitt per il calcolo del tempo di movimento.

Legge di Fitt: MT = a + b * log2 (2D/W)

con a definito come il tempo in secondi necessario per iniziare o smettere di muoversi, b la velocità del dispositivo, D la distanza dal punto iniziale al centro dell'obiettivo, W la larghezza dell'obiettivo misurata lungo l'asse su cui ci si sta muovendo.

Si danno i seguenti criteri per la classificazione di dispositivi di puntamento:

- Sulla base del tipo di input:
  - Diretto : se il puntatore si trova nella stessa posizione fisica dell'utente
  - Indiretto : quando traduce il movimento sullo schermo
- Sulla base del modo in cui il movimento viene mappato sull'interfaccia:
  - Assoluto : quando il mapping tra lo spostamento nel mondo fisico viene replicato così com'è sul dispositivo
  - Altrimenti
- Sulla base di come i dispositivi producano il segnale sulla bse dello spostamento:
  - Isotonico : si può muovere nello spazio e misura lo spostamento
  - Isometrico : è fisso e misura la forza che viene applicata
  - Elastico : la forza applicata è proporzionale allo spostamento
- Sulla base della velocità con cui si fa avanzare il puntatore:
  - Position control : ci controlla la posizione del puntatore
  - Rate control : ci controlla la velocità e direzione del puntatore

I dispositivi di puntamento innovativi sono l'eye tracker che misurano la posizione della pupilla e i movimenti degli occhi. I vari metodi per estrarre informazioni sono:

- Bright-pupil : si illumina con luca infrarossa l'occhio e si misura la posizione della pupilla. La luce è coassiale all'occhio in modo che la pupilla rifletta la luca e illumini di rosso l'occhio.
- Dark-pupil : si illumina con luca infrarossa l'occhio e si misura la posizione della pupilla. La luce non è coassiale all'occhio e la pupilla appare nera.
- Passive light : si misura la posizione della pupilla con la luce ambientale Tutto ciò è differente dal gaze tracking che si occupa di capire dove il soccetto stia guardando riportando l'angolo della pupilla nello spazio tridimensionale in cui si trova l'utente. Ma per rendere l'eye tracking un sistema di puntamento va affiancato con il gaze tracking. I dispositivi dataglove sono dei guanti che permettono di rilevare i movimenti delle mani e delle dita. I dispositivi aptici sono dispositivi che permettono agli utenti di interagire con ambienti virtuali tramite feedback tattili. Altri dispositivi sono gli smart paper e le lavagne digitali.

HID suoni

In fisica, un suono è una vibrazione che si propaga attraverso una onda acustica, tramite un gas, liquido o solido. Il suono può essere catturato usando microfoni, device con sensori che convertono suoni in segnali elettrici. I microfoni possono ossere usati anche in tandem, formando un array di microfoni in modo di estrarre precisamente il suono che vogliamo evitando suoni indesiderati, come il rumore di fondo, e riconoscere la direzione del suono.

HID video

I sensori di immagini sono sensori che rilevano attraverso le radiazione luminose e convertono immagini in informazioni. Ci sono due tipi di sensori di immagini: i charge-coupled device(CCD) e gli active-pixel sensor(CMOS). Entrambi i sensori sono basati su metal-oxide-semicoductor(MOS). Il 3D scanner consiste nel rappresentare dimensione e posizione nello spazio tridimensionale di oggetti e ambienti. I 3D scanner sono divisi in due categorie:

- Passivi : non emettono radiazioni elettromagnetiche, si affidano all'illumazione ambientale. Alcuni esempi:
  - Camere stereoscopiche : usano due camere poste ad una distanza focale simile a quella dell'occhio umano e, calcolando la differenza nelle immagini pixel per pixel, creano una terza immagine detta depthmap per creare l'illusione di profondità
  - Sistemi fotometrici : assumono l'esistenza di una sorgente luminosa controllabile e analizzando gli spostamenti delle ombre al variare dell'incidenza (della sorgente luminosa) vengono calcolati sia la profondità sia il colore.
  - Tecniche silhouette : tipicamente messe a disposizione anche da applicazioni per dispositivi mobili, scattando più foto dello stesso oggetto da diversi angoli e usando l'accelerometro e la piattaforma iniziale posizionano i piani immagini nello spazio tridimensionale e ricostruiscon l'oggetto.
- Attivi : emettono radiazioni elettromagnetiche, tipicamente luci, ultrasuoni o raggi x. Alcuni esempi:
  - Time-of-flight : inviano un impulso laser e misurano il tempo di ritorno
  - Triangolazione : inviano un impulso laser, usando l'angolo di riflessione e il disallineamento ottico tra emettitore e ricevitore, permettono di ricostruire la distanza punto-punto.
  - Scanner a luce strutturata : viene proiettata un'immagine geometrica nota su un oggetto tridimensionale e vengono analizzate le deformazioni
  - Scanner a luce modulata : viene proiettata una luce modulata su un oggetto tridimensionale e vengono analizzate le deformazioni

Un Inertial Measurement Unit (IMU) è un dispositivo elettronico che misura e reporta l'accelerazione, la velocità angolare e l'orientamento di un corpo rigido. Un IMU è composto da un accelerometro, un giroscopio e a volte un magnetometro.

I dispositivi indossabili sono device che possono essere indossati dall'utente nei quali l'interfaccia e l'unità computazionale sono uniti e spesso sono specializzati nella raccolta di certi dati biometrici o di movimento. Un dispositivo heart rate monitor permette di misurare il battito cardiaco dell'utente utilizzando dei sensori PPG che tipicamente illuminano la pelle e analizzano la luce riflessa. Invece i sensori ECG usano dei sengali elettrici per misuare l'espanzione e la contrazione del cuore. Un dispositivo EEG headset permette di monitorare gli impulsi elettrici del cervello posizioando elettrodi non-invasivi sullo scalpo dell'utente.

Natural User Interface (NUI)

La NUI è un'interfaccia utente naturale e semplice da usare con interazione diretta e consistente con il comportamento naturale. Una interfaccia utente naturale è una interfaccia che è effettivamente invisibile,e resta invisibile mentre l'utente impara progressivamente ad avere interazioni via via più complesse. Ci sono dei gesti di tipo touch che oramai fanno parte del background culturale, come il tocco, la lunga pressione, lo scroll, il pinch e lo swipe. Sono diventati intuitivi per gli utenti poiché le

interfacce restituiscono il feedback associato molto velocemente. La parola “naturale” viene usata in riferimento al goal di progettazione delle interfacce: si ambisce a offrire un’esperienza che non richieda la comprensione di come una macchina funzioni; si punta a un’interfaccia che si comporti come un oggetto fisico

Poichè una NUI sia considerata tale si richiedono:

- Apprendimento progressivo
- Expertise instantanea
- Interazione diretta
- Basso carico cognitivo

Una strategia per realizzare NUI è l'uso della Reality User Interface (RUI), anche conosiuta come Reality-Based Interface (RBI). Un esempio di RUI è l'uso di device indossabili per rendere clickabili oggetti del mondo reale. Invece un esempio di NUI non basata su RBI è limitare le funzionalità e le personalizzazioni in modo che gli utenti abbiano ben poco da imparare.

Graphical User Interface (GUI)

Descrive l'organizzazione di un interfaccia a livello di navigabilità. Si distinguono le seguenti strutture:

- Struttura gerarchica : l'utente comincia a navigare da una pagina principale e poi naviga in profondità
- Struttura sequenziale : l'utente naviga passo per passo seguendo un percorso prestabilito. La struttura ha un ingresso ed almeno un uscita
- Struttura matriciale : l'utente non viene vincolato a nessuna struttura, è permessa una navigazione molti a molti. Spesso vengono messi dei navigator per spostarsi all'interno del sistema.
- Struttura a database : rappresentazioni di dati in tabelle

L'architettura dell'informazione spiega come i contenuti che si vogliono presentare all'utente sono organizzati, senza veicolare il modello concettuale.

Le componenti principali sono:

- Schemi o strutture organizzative : come viene organizzata e strutturata l'informazione
- Sistemi di labelling : come viene rappresentata l'informazione
- Sistemi di navigazione : come gli utenti si spostano all'interno del sistema
- Sistemi di ricerca : come gli utenti possono cercare informazioni

Si distinguono 2 tipi di schemi organizzativi:

- Esatti : organizzano le informazioni in modo oggettivo in sezioni mutamente esclusive. Ad esempio:
  - Schema alfabetico : fa uso dell'ordinamento lessicografico
  - Schema cronologico : organizza il contenuto sulla base della data
  - Schema geografico : organizza il contenuto sulla base della posizione
- Soggettivi : organizzando le informazioni in modo diverso per ogni utente. Ad esempio:
  - Topic scheme : organizza il contenuto in base all'argomento
  - Task scheme : organizza il contenuto sulla base dei bisogno, delle azioni, delle domande o dei processi che portano un utente a quel contenuto
  - Audience scheme : organizza il contenuto sulla base del segmento di utenza
  - Metaphoric scheme : organizza il contenuto legandolo a concetti familiari

È anche possibile combinare più schemi organizzativi per ottenere una struttura ibrida.

Nell'architettura dell'informazione è importante anche il design della navigazione, cioè come gli utenti si orientano nell'interfaccia. I principi base della navigazione sono la findability e la discoverability. Per findability intendiamo l'essere abile di trovare l'informazione che stavi cercando mentre per discoverability ci riferiamo alla possibilità degli utenti di scoprire nuove informazioni o feature che non stavano cercando.

Quando un utente inizia un interazione con il sistema è influenzato dalle varie informazioni che ha a disposizione. Ci sono quindi una serie di pattern, cioè una serie di eventi in successione, che si ripetono:

- Quit : l'utente cerca l'informazione, vede il risultato ed esce
- Narrow : l'utente cerca l'informazione, vede il risultato e cerca di restringere la ricerca tramite strumenti di filtraggio
- Expand : l'utente cerca l'informazione, vede il risultato e cerca di avere più informazioni
- Pearl growing : l'utente cerca l'informazione, apre uno dei risultati e utilizza i collegamenti al suo interno
- Pogosticking : l'utente cerca l'informazione e poi ripete l'azione di aprire e chiudere i vari risultati
- Trashing : l'utente cerca l'informazione, guarda i risultati, e poi torna a fare una nuova ricerca aggiungendo dettagli
- Berry picking : l'utente cerca l'informazione, apre un risultato e torna a fare una nuova ricerca aggiungendo dettagli

Alcuni di questi comportamenti possono accadere a volte, ma quando capita troppo spesso è un segnale che l'architettura dell'informazione non è ben fatta e si tratta di anti-pattern.

In parallelo con i pattern di comportamento ci sono anche dei design pattern che migliorano l'architettura dell'interfaccia.

- Autocomplete : completa la ricerca dell'utente con suggerimenti
- Autosuggest : consiglia termini di ricerca simili
- Instat result : risponde alla ricerca mentre l'utente la sta ancora scrivendo
- Did you mean : dopo che l'utente ha scritto la ricerca, il sistema gli chiede se intendeva un'altra cosa
- Autocorrect : invece di suggerire termini simili, corregge la ricerca dell'utente automaticamente
- Best first : ordina i risultati in base a un criterio dato da un algoritmo
- Partial matches : mostra i risultati che si avvicinano di più alla ricerca dell'utente
- Related seraches : mostra ricerche correlate per possono ispirare l'utente
- Federated search : da la possibilità di cercare in più fonti di informazione
- Faceted navigation : da la possibilità di filtrare i risultati in base a delle categorie
- Advanced search : da la possibilità di fare ricerche avanzate con più parametri
- Scoped search : se è possibile divide i risultati in categorie
- Personalization : adatta i risultati in base al profilo dell'utente
- Pagination : mostra un numero massimo di pagine
- Structurable results : mostra i risultati tramite strutture che sono più congrutenti con i contenuti
- Actionable result : dipendentemente dal tipo di contenuto, mostra azioni che l'utente può fare
- Comparing result : da la possibilità all'utente di fare confronti tra i risultati
- Unfied discovery : mostra i risultati in base a più fonti di informazione

Il Document Object Model è una rappresentazione in un linguaggio di alto livello della struttura di un sito web o di una app che viene interpretata e renderizzata dal browser, è tipicamente organizzato come un documento gerarchico. Quando una pagina web JavaScript è caricata, viene creato un Document Object Model (DOM) della pagina che è una rappresentazione object oriented del documento HTML che si interfaccia tra JavaScript e il documento dando la possibilità di:

- Aggiungere, cambiare e rimuovere elementi HTML
- Cambiare stili CSS
- Reagire agli eventi
- Creare nuovi eventi

Gli elementi dell'interfaccia sono:

- Elementi di input
- Elementi di navigazione
- Componenti informativi
- Container

UX design

Per portare avanti un progetto di un prodotto è necessario curare anche l'esperienza utente, cioè l'esperienza che l'utente ha quando interagisce con il prodotto e per farlo al meglio è necessario seguire un processo di UX design che ci permette di capire quali sono gli utenti,quali sono i loro comportamenti e quali sono i loro bisogni. Ciò si ottiene seguendo delle fasi dinamiche e alternabili.

Per identificare le personas, cioè l'archetipo di uno dei possibili utenti possiamo usare le tecniche di:

- Task analysis : analisi delle attività che l'utente deve svolgere
- Feedback : analisi dei feedback che l'utente da durante l'utilizzo del prodotto
- Prototipazione : creazione di prototipi per testare le interazioni con l'utente

Dipendentemente dai dati che abbiamo possiamo avere 3 tipi di personas:

- Proto-personas : un prototipo di persona ideale creata sulla base di dati aneddotici
- Qualitative personas : creata sulla base di ricerche su un campione medio-piccolo di utenti
- Statistical personas : creata collezionando e analizzando dati da una grande quantità di utenti.

Il principio di Pareto ci consiglia di concentrarci sulle personas che rappresentano il 20% degli utenti che portano il 80% del utilizzo complessivo del prodotto.

Le informazioni di cui teniamo conto di una personas sono:

- Demografiche
- Personali
- Attitudinali e cognitivi
- Obiettivi e motivazioni
- Comportamentali

Le personas verranno poi definiti con qualcosa di più astratto chiamato archetipo che rappresenta un tipo di utente generico con le sue caratteristiche.

Un requirement è un servizio o una caratteristica che soddisfa un bisogno di un utente, come ad esempio funzioni, vincoli, regole aziendali o altri tipi di elementi di cui il prodotto deve essere dotato per soddisfare le esigenze degli utenti. È più facile capire i requirements corretti se abbiamo ben presente le personas. Esistono 2 tipi di requirements:

- Funzionali : descrivono quali funzioni deve avere il prodotto
- Non funzionali : specificano i tratti qualitativi del prodotto

Una user stories è una breve dichiarazione o astrazione che identifica l'utente e il suo bisogno/obiettivo. É un requisito espresso dalla prospettiva di un dell'obiettivo dell'utente. aiutano a documentare informazioni pratiche riguardo gli utenti e aiutano gli sviluppatori a tracciare una roadmap. struttura: As a 'role', I want 'feature' because 'reason'. tutti possono scrivere user stories ad ogni livello di dettagli. i dettagli possono essere aggiunti splittando le user stories in multiple user stories o aggiugendo condizioni di soddisfazione.

Uno scenario è una situazione che cattura come gli utenti interagiscono con un prodotto. Un buono scenario deve rispondere:

- Chi è l'utente?
- Motivazione e aspettativa dal prodotto?
- Qual'è il suo obiettivo?

Grazie agli scenarios possiamo determinare:

- I punti importanti durante progettazione per l'UX
- Fasi del processo che richiedono ulteriore revisione e attenzione
- Le principali esigenze e motivazioni dell'utente

I metodi principali per scrivere scenarios sono:

- Goal o task orinentati agli scenarios
- Elaborated scenarios
- Full scale task scenarios

Gli use cases sono la naturale evoluzione degli scenarios. Consistono della completa narrativa di quali azioni l'utente compie per svolgere uno scenario. Ogni caso d'uso è rappresentato come una sequenza di passaggi che iniziano con l'obiettivo dell'utente e terminano quando l'obiettivo è raggiunto. Un caso d'uso aggiunge valore perché aiuta a spiegare come il sistema dovrebbe comportarsi e forniscono una lista di obiettivi.

La differenza tra scenarios e caso d'uso è ch uno scenario richiede una situazione che può avere uno o più attori che intraprendono una determinata funzionalità. un caso d'uso coinvolge un attore e il flusso che un particolare attore prende in una determinata funzionalità o percorso. La differenza principale è la prospettiva.

Gli uses case includono:

- L'utente
- Cosa vuole fare
- Il suo scopo
- Step necessari per raggiungere lo scopo
- Feedback
- Trigger
- Basic flow
- Alternative flow

Mentre non includono:

- Dettagli implementativi o di scelta tecnologica
- Dettagli di UI

I passaggi da seguire per la creazione di un caso d'uso sono:

1. Identificare le personas
2. Sceglierne una per caso d'uso
3. Identificare il suo scopo
4. Discenderne i task principali da quelli secondari
5. Considerare le sequenze alternative
6. Accoppiare punti in comune tra in vari casi d'uso
7. Ripetere per tutte le personas

Front-end design Wireframing

wireframe: per trasmettere la tua idea alle altre persone, si creano progetti che aiutano la comuncazione tra designer e sviluppatori attraverso sketch basici che mostrano la struttura generale.

Tipi di wireframe:

- Low fidelity: semplici sketch che si concentrano sulla struttura
- High fidelity: rappresentazioni dettagliate con una basica interfaccia e icone.

Il contesto ha un importante ruolo nell'UX design per adattarci ai comportamenti e alle preferenze degli utenti.

L'adattibilità è la capacità di un sistema di adattarsi a diversi contesti senza cambiare la sua struttura e copre le nozioni di:

- Responsive design
- Accessibilità

Il responsive design è un approccio di progettazione che permette ai siti web di adattarsi a diversi dispositivi e dimensioni dello schermo per avere una buona esperienza coerente con tuti i dispositivi.

Le tecniche di responsive design sono:

- Flexible grid
- Flexible images
- Media queries & breakpoints

Una filosofia di reposive design è mobile first che consiste nel progettare prima per i dispositivi mobili e poi per i desktop e ciò ci fa concentrare prima sulle funzionalità base. Ciò si oppone alla degradazione in cui si sviluppa prima per i desktop per poi rimuovere alcune funzionalità per adattarsi ai dispositivi mobili perdendo però consistenza dei vari dispositivi.

Altre best practise del responsive design:

- Keep it simple : design minimal
- Priorizzare il contenuto
- Progettare per il touch : bottoni grandi
- Ottimizzare tempi di caricamento : ridurre componenti pesanti come immagini
- Testare su diversi dispositivi

Adaptive design è un altra alternativa al responsive design che consiste nel creare un interfaccia che si adatta automaticamente ai vari utenti in modo fluido e flessibile. Solitamente realizzata creando più versioni del sito web per ogni dispositivo.

Adaptive UX ottiene informazione basiche tra i vari utenti e le usa per adattare l'interfaccia e consigliare ad ogni utente cosa è migliore per lui attraverso il collaborative filtering e il content based filtering.

Nell UX design è cruciale anche l'accessibilità che consiste nel rendere il prodotto accessibile a tutti gli utenti indipendentemente dalle loro capacità fisiche o cognitive in modo che ogni utente possa accedere a tutte le funzionalità del prodotto. Alcuni problemi di accessibilità sono:

- Visuale
- Motoria
- Uditiva
- Convulsioni
- Cognitive

L'azienda W3C ha posto degli standard e delel linee guida per l'accssibilità chiamate Web Content Accessibility Guidelines (WCAG) e consigliano:

- Alternative testuali per contenuti non testuali come immagini e grafici
- Sottotioli o altre alternative per contenuti multimediali
- Diversi modi per usufruire dei contentuti
- Contenuti facili da vedere e da ascoltare
- Utenti possono usare altre modalità di input oltre la tastiera
- Struttura facile da navigare e da capire

Oltre ai wireframes che ci aiutano a comunicare le nostre idee, abbiamo bisogno di uno schema di navigazione che ci aiuti a capire come gli utenti possono navigare nel nostro sito web. Lo facciamo attraverso gli user flows che sono una rappresentazione visuale di come gli utenti possono navigare nel nostro sito web. I più comuni sono:

- Flowchart è un diagramma di flusso che mostra le varie possibili scelte che un utente può fare e le conseguenze di queste scelte.
- Wireflows esprimono un diagramma di flusso usando wireframe invece di descrizioni astratte

Metodi e strumenti per l'innovazione

Un innovazione è qualcosa di originale e utile che entra nel mercato e che cambia il modo in cui le persone vivono e lavorano. Spesso è legata al mondo delle invenzioni ma non necessariamente.

Sustaining innovation cioè migliorare un prodotto esistente:

- Step by step
- Basso rischio
- Bassa velocità
- Non cambia organizzazione aziendale
- Non ha bisogno di nuove competenze da parte degli utenti
- Basse probabilità di scalare il mercato
- Target di mercato stabile

Disruptive innovation, usato per creare nuovi mercati cerca di non basarsi sulle tecnologie esistenti per raggiungere utenti che non sono serviti dai prodotti esistenti

Lo human centered desing process cerca di sviluppare un sistema che sia utile e utilizzabile concentrandosi sugli utenti cercando di migliorare l'efficacia e l'efficienza.

Il product management si occupa del cosa mantre il product development si occupa del come. Lavorano insieme, il product development team prende le specifiche dal product manager e le implementa in un prodotto funzionante.

Ideo ha sviluppato un processo di innovazione human centered che si basa su 3 fasi:

1. Ispirazione : approfondisce i bisogni e le richieste degli utenti per migliorare uno strumento, osservando come viene usato.
2. Ideazione : si cerca di interpretare le conoscenze assunte per arrivare a qualcosa di più tangibile tramite la creazione di un semplice prototipo. non definitivo e neacnhe perfetto ma utile come punto di partenza per continuare a testare e fare considerazioni.
3. Implementazione : si cerca di capire come il prodotto può essere implementato nel mondo reale e come può essere scalato.

Ciascuna di queste fase è svolta dal team con un approccio a fisarmonica, seguendo uno schema detto "Double Diamond" in cui all'inizio si lavora per produrre idee e soluzioni in quantità (divergenza) per poi concentrarsi su quelle più promettenti (convergenza).

Il design thinking è un approccio all'innovazione che poggia le sue fondamenta sulla capacià di risolvere problemi complessi utilizzando una visione e una gestione del progetto basata sulla creatività. Si basa su un processo iterativo che cerca di capire l'utente facendo delle assunzioni.

Gli obiettivi sono:

- Avvicinarsi al cliente
- Favorire la creatività e generare idee
- Sperimenare le idee con prototipi

Le fasi principali sono:

1. Empatizzare : identificare il problema e l'obiettivo, scrivendo le personas
2. Definire : delineare i bisogni degli utenti e le varie categorie di utenti, facendo user stories
3. Ideare : ricercare una soluzione al problema, facendo brainstorming
4. Prototipare : realizzare i primi prototipi
5. Testare : testare i prototipi sul campo e raccogliere feedback dagli utenti

Lo Human centered desing è un mindset, un modo di pensare e di approcciarsi allo sviluppo di prodotti, il Design thinking invece è un vero e proprio metodo di lavoro organizzato per fasi che consente di sviluppare prodotti centrati sull'utenze grazie alle tecniche orientate alla stimolazione della creatività e alla produzione di idee. Human centered design e Design thinking sono compatibili e combinabili in un metodo detto Social Enterprise Thinking

Metodi di sviluppo per prodotti innovativi

Il metodo di sviluppo classico è sicuramento il modello waterfall, una metodologia lineare per lo sviluppo di progetti in cui ogni fase ricasca sulla successiva ed è strutturato in 5 parti:

1. Analisi dei requisiti
2. Progettazione
3. Svilluppo
4. Collaudo
5. Manutenzione

Dal modello waterfall sono poi nati altri modelli. Il modello agile è un metodo di sviluppo software che si basa sul creare e rispondere al cambiamento e pone le fondamenta sui 12 principi del manifesto per sviluppo software agile Una cosa che contraddistingue agile è il fatto di concentrarsi sulle persone che fanno il loro lavoro e come lavorano insieme. Le soluzioni vengono raggiunte attraverso la collaborazione tra team multidisciplinari e auto-organizzati.

Il modello scrum è una sottocategoria agile che utilizza un metodo interattivo ed un apporccio incrementale per ottimizzare la prevedibilità e per controllare il rischio. Il modello è particolarmenete utile per progetti complessi e con un piccolo numero di persone e funziona sul dividere il progetto in blocchi rapidi per raggiungere piccoli obiettivi alla volta chiamati sprint che non sono più lunghi solitamente di due settimane. Il team traccia i propri progressi in meeting giornalieri di 15 minuti chiamati daily scrum. Un aspetto forte dello scrum è la possbilità di cambiare idea e di adattarsi ai cambiamenti del mercato o alle emergenze poiché il team non si pone il problema di pensare a tutto fin dall'inizio ma si concentra su piccoli obiettivi alla volta. Ogni sprint inizia con una pianificazione degli eventi e stabilendo gli obiettivi e termina con una revisione di ciò che è stato fatto nel periodo prestabilito, adattando il prossimo sprint in base ai risultati ottenuti. Ci sono 3 ruoli principali in scrum:

- Product owner : responsabile del prodotto e del suo sviluppo
- Scrum master : responsabile del processo e del team
- Team : responsabile dello sviluppo del prodotto

Il modello devops è sottocategoria agile per cloud che ha la necessita di avere una gestione più flessibile, affidabile, sicura e contollabile gestione dei rilasci con cicli di rilascio più brevi e frequenti.

Prototipazione di interfacce

Ogni anno le compagnie lanciano migliaia di nuovi prodotti di tutti i tipi nel mercato, ognuno dei quali seguito da un team che sperano nel suo successo, ma la maggior parte di questi prodotti fallisce, circa l'80% fallisce. Come nella legge criminale, una persone è innocente finche non vengono trovate prove a suo carico, un prodotto è un fallimento finche non viene dimostrato il contrario. L'unico metodo per combattere la legge del fallimento del mercato è quello di testare oggettivamente le idee prima di investire tempo e denaro in un prodotto che potrebbe non avere successo. Il pretotyping ci offre strumenti e tecniche di cui abbiamo bisogno per validare un idea con il minimo delle risorse e in tempi brevi.

Lost in Translation Problem: Un idea è un astrazione che immagini nella tua testa. Nel momento in cui provi a comunicarla ad un'altra persona si ha un problema di

traduzione, specialmente quando la tua idea è differente da quella dell'altra persona.

Prediction Problem: anche se l'audience capisce la tua idea, non è detto che la capisca allo stesso modo in cui la capisci tu. Ognuno ha la propria esperienza e la propria visione del mondo e questo può portare a fraintendimenti.

I falsi positivi sono un problema perchè ti fanno credere che il tuo prodotto sia un successo e ti fanno perdere soldi e tempo mentre i falsi negativi ti fanno credere che il tuo prodotto sia un fallimento e ti fanno perdere un'opportunità. Per minimizzare i falsi positivi e i falsi negativi bisogna collezionare dati oggettivi e l'unico modo per farlo è quello di trasportare le proprie idee da thoughtland ad actionland. Nella thoughtland usi idee astratte su cui ti poni domande e ricevi opinioni. Nella actionland usi artefatti concreti su cui svolgi azioni e collezioni dati.

Un prototipo ti aiuta a fallire più velocemente ma spesso non velocemente abbastanza e comunque con un costo non trascurabile. Tra un'idea astratta e un prototipo c'è un'area grigia in cui si può fare pretotyping. I pretotipi ci danno la possibilità di collezzionare dati per il mercato che ci aiutano a decidere se iniziare ad investire sull'idea. Un pretotipo è un mock-up di un prodotto o servizio che può essere costruito in poco tempo.

Gli obiettivi del pretotyping sono aiutare a:

- Identificare funzionalià chiave
- Decidere le funzionalità da implemtare nel mockup
- Test sui mockup e collezione dei feedback e dati
- Analisi dei dati e determinare il prossimo passo

I 7 pilasti del pretotyping:

- Obbedire alla legge del fallimento del mercato
- Assicurarsi di star costruendo il prodotto giusto
- Non perdersi in chiacchere, idee o opinioni
- Fidarsi solo dei propri dati
- Fare pretotyping
- Parlare con i numeri e con i fatti
- Pensa globalmente, testa localmente

Flusso del pretotyping

1. Isolare l'assunzione chiave : qual è l'assunzione o funzionalià chiave che se non funziona il prodotto fallisce?
2. Scegliere un tipo di pretotype : qual è il tipo di pretotyping che permette
3. Fare ipotesi di mercato (ipotesi XYZ)
4. Testare il pretotype : mettere il pretotipo nel mondo reale e vedere quante persone sono interessate e quante ci interagiscono
5. Imparare, rifinire, hyperzoom : valutare i risultati, rifinire il pretotipo con i nuovi dati e se l'ipotesi ha retto decidere altre situazioni in cui testare il pretotipo per avere una visione completa.

Tipi di pretotyping:

- Fake door : un marketing entry point per un prodotto che non esiste ancora e può essere utliizzato per pubblicizzare un servizio non ancora pronto e misurare l'interesse degli utenti. Serve per capire se l'oggetto che si vuole sviluppare può avere successo e risparmiare tempo e denaro. Si può utilizzare questo pretotyping qunado l'idea può essere descritta in poche e semplici parole, senza possedere nulla di fisico.
- Mechanical turk : un servizio che permette di simulare e trasmettere l'esperienza del prodotto finale ad un utente senza che esso sia sviluppato per mostrare la reale esperienza che l'utente avrà con il prodotto.
- Impersonator : è un pretotipo che riesce a far sperimentare un'esperienza all'utente in modo estremamente economico e con lavoro minimo dietro.
- Pinocchio : è un pretotipo chiaramente falso che serve per veicolare un messagio così distante dalla realtà che è faticoso e difficile da spiegare in altri linguaggi naturali, usato spesso per testare l'interesse e il possibile uso di prodotti innovativi non ancora lanciati da nessuno, nemmeno in maniera simile, su mercato.
- One night stand : una tecnica di veicolazione di un pretotipo che consiste in un market test che offre solo l'esperienza senza alcun altro tipo di infrastruttura in una finestra di tempo limitata
- Facade : è un impersonator ma è usato per dare un'immagine dell'azienda e non del prodotto stesso e viene spesso usato per promuovere servizi.

Dopo aver superato le varie fasi di pretotyping ed aver accumulato una sicurezza sufficiente per il successo del prodotto, lo step successivo è quello di produrre il Minimum Viable Product cioè la versione minimale del prodotto contenente solo ed eslusivamente le features che si sono pretotipate attraverso la fase precedente. Non si hanno le mani sul prodotto definitivo ma su qualcosa di vendibile, in modo da ottenere del ricavo e dell'utile che se sufficiente permetterebbe la produzione del prodotto finale.

Prototipi e Mock-Ups

Un mock-up è una rappresentazione con alta fedeltà del prodotto che include la maggior parte dei dettagli visibili e gli elementi di design del wireframe. Quindi i mock-ups possono essere usati per rifinire e far notare gli elementi visuali di design ma peccano di funzionalità e perciò non sono adatti a fare testing. I mock-ups vengono usati dai designer per comunicare meglio i loro concetti di design e le funzionalità in modo più efficiente ai resposabili e ai clienti.

Un prototipo è un'implementazione incompleta o approssimativa di un prodotto su cui è possibile interagire prima di rilasciare il prodotto finale. C'è sempre bisogno di un prototipo poiché alla prima prova di prodotto si trovano spesso molti problemi e ciò ci permette di testare e rifinire il prodotto prima di rilasciarlo. I vantaggi di creare un prototipo sono:

- Creare delle solide fondamenta per l'ideazione del prodotto finale e dare al resposabile una chiara idea dei potenziali benefici, rischi e costi.
- Cambiamenti possono essere fatti in modo più economico e veloce
- I feedback degli utenti aiutano ad indentificare gli aspetti su cui lavorari
- La prototipazione ci da la possibilità do sperimentare per le necessità e i problemi degli utenti in modo da migliorare
- I responsabili si legano più facilmente al prodotto
- Time-to-market migliorato grazie alla riduzione degli errori prima del rilascio

Un prototipo assomiglia al prodotto finale in vari gradi di fedeltà che solitamente sono 3:

- Low-Fidelty Prototypes : solitamente fatti con carta e penna, connette i vari low-fidelty wireframes, molto veloce da fare e facile da modificare. Non molto realistico e solitamente sono versione molto acerbe semplificando molto il prodotto finale e le sue funzioni.
- High-Fidelity Prototypes : prototipi molto più avanzati, solitamente sviluppati attraverso l'uso di software specifici, danno la sensazione di prodotto finito e sono testabili. Hanno bisogno di più tempo e costi per essere creati e modificati.
- Mid-Fidelity Prototypes : sono un compromesso tra i due prototipi precedenti, sono più realistici dei low-fidelity ma più veloci da creare e modificare dei high-fidelity.

Dimensioni di fideltà di un prototipo:

- Realismo : quanto il prototipo è simile al prodotto finale

- Scope : ampiezza e profondità del prototipo
- Funzionalità : quali e come le funzionalità sono implementate
- Dati : se il prototipo funziona su dati reali o simulati
- Autonomia : come il prototipo può operare senza il bisogno dell'utente
- Piattaforma : se il prototipo è un implementazione provvisoria o definitiva

In particolare lo scope ti da informazioni su quanto buono potrà essere il prototipo. Esistono due tipi di scope:

- Orizzontale : ci da un visione d'insieme del sistema, non andando nei dettagli di ogni feature
- Verticale : ci da una visione approfondita di una singola feature

Tecniche di prototyping:

- Paper Prototypes : è un low-fidelity prototype, creato su carta e rappresentano il layout basico e le funzionalità di un prodotto o servizio, rimanendo una veloce e senza costi via per testare l'interfaccia utente e rifinire prima di investre tempo e risorse in un prototipo più avanzato.
- Wireframes Prototypes : sono prototipi digitali che rappresentano il layout e le funzionalità di un prodotto o servizio, senza molti dettagli visivi. Richiede creare sketch del layout e delle funzionalità base usando semlici forme e simboli che rappresentano i vari elementi dell'UI
- Wizard of Oz Prototypes : sono prototipi che simulano il comportamento dei prodotti o servizi usando input umani invece delle tecnologie e vengono spesso utilizzati quando queste tecnologia non sono ancora sviluppate per capirne la reale utilità. ciò richiede spesso più persone per simulare che svolgono interazioni dirette.
- Functional Prototypes : sono prototipi che includono funzionalità reali del prodotto e sono usati per rifinire gli aspetti tecnici del prodotto. Richiedono la costruzione di prototipi con codice e hardware o strumenti che danno la possibilità di interagire con gli elementi. Sono particolarmente utili per gli stage più avanzati di sviluppo del prodotto.
- Non-Functional Prototypes : sono prototipi che semrano esattamente come il prodotto reale ma non hanno funzionalità reali e sono usati per testare il design e l'esperienza utente del prodotto prima di investire ulteriori risorse. Un po' di codice sarà comunque necessario ma senza andare nel dettaglio. Possono essere usati per testare sia l'esperienza dell'utente oppure per testare il design visuale del prodotto.

Wireframes vs Mockups vs Prototypes

Le persone spesso fanno confusione tra i vari concetti. I prototipi sono versioni più avanzate dei mock-ups che includono funzioni di navigazione e funzioni. I designers solitamente usano i mock-ups e i wireframes per creare il prototipo finale. I prototipi sono quindi usati per fare testing prima di muoversi nella fase finale di sviluppo.

Se è clickabile e interattivo oppure risponde alle azioni in altro modo è un prototipo funzionale, altrimenti se ha design, colori e immagini è un mock-up, altrimenti è un semplice wireframe.

Industria 4.0

Quando pensiamo al design di un prodotto connesso, cerchiamo di concentrarci sugli elementi più visibili e tangibili, l'industrial design e la user interface. Ma ci sono maggiori preoccupazioni che impattano la user experience. Possiamo avere una bellissima interfaccia ma l'utente potrebbe avere comunque una cattiva esperienza. Per questo è importante considerare anche l'industrial design, l'interazione tra i dispositivi e l'interazione tra l'utente e il dispositivo.

Con l'internet of things (IoT) i prodotti sono provvisti di proprio identificatore sono connessi tra loro e con il mondo esterno, questo permette di creare nuove esperienze e nuovi servizi.

Negli ultimi secoli l'industria si è sempre evoluta e ha avuto molte fasi:

- Industria 1.0 : meccanizzazione con utilizzo di energia a vapore
- Industria 2.0 : produzione in serie con utilizzo di energia elettrica
- Industria 3.0 : automazione con utilizzo di elettronica e informatica
- Industria 4.0 : smart factory

L'industria 4.0 descrive l'organizzazione di un sistema di produzione intelligente che si basa tecnologie e dispositivi che interagiscono tra loro e una fabbrica intelligente in cui i sistemi computerizzati monitorano processi fisici creando una copia del mondo fisico e prendendo decisioni in autonomia. In futuro, le aziende di successo utilizzeranno l'Internet of Things per ottenere una nuova crescita attraverso tre approcci: incrementare i ricavi aumentando la produzione e creando nuovi modelli di business ibridi, sfruttare le tecnologie intelligenti per alimentare l'innovazione e trasformare la loro forza lavoro.

Un digital twin è una replica digitale di un'entità fisica che vengono collegate con un ponte che gli fa scambiare dati ma esistere simultaneamente.

Un prodotto smart è un prodotto fisico che ha un servizio digitale al suo centro.

Fare design per un prodotto delle Internet of Things è interamente più complesso del design di un servizio web.

Fare il design di un prodotto IoT richiede un approccio multidisciplinare dato che ci sono vari aspetti del design da considerare non immediatamente visibili. UX per IoT richiede una natura specializzata, un abilità di connettere mondo digitale con mondo fisico e il fatto che i prodotti IoT sono distribuiti su più dispositivi diversi. Molte cose dell'IoT sono dispositivi informatici integrati e specializzati, che differentemente dai computer, i loro software e hardware sonon ottimizzati e compiono solo specifiche funzioni. La loro UI non è semplicemente a livello software ma spesso è estesa con schermi e bottoni per l'uso di vari controlli e ciò cambia le interazioni che hanno gli utenti con il device che non hanno più il beneficio delle guide che avevano sul web.

Nuovi principi del design

- UX per IoT
- Device specializzati con differenti capacità
- Far interagire i dispositivi tra loro
- Controllo remoto
- Design per network
- Design per risparmio d'energia

Usability Testing

Quando abbiamo una sorta di prototipo, dovremmo fare testing con gli utenti. Questo processo ci permette di valutare il prodotto o il servizio tramite il giudizio diretto degli utenti interessati. In questo modo possiamo indivisuare varie problematiche da risolvere, capire cosa migliorare e cosa gli utenti preferiscono.

L'usabilità è un attributo qualitativo che ci indica quanto è facile l'interfaccia utente da usare, ed è definita in 5 componenti:

- Apprendibilità : quanto è facile per gil utenti completare compiti basilari la prima volta che usano l'interfaccia
- Efficienza : velocità con cui gli utenti completano compiti dopo che hanno imparato l'interfaccia
- Memorabilità : quanto è facile per gli utenti riprendere a usare l'interfaccia dopo un periodo di non utilizzo

- Errori : quanti errori fanno gli utenti, quanto gravi sono e come possono recuperare
- Soddisfazione : quanto piace agli utenti usare l'interfaccia

Nell'UX design, un altro concetto chiave è l'utility che si riferisce alle funzionalità e alle caratteristiche di un prodotto o servizio che soddisfano i bisogni degli utenti.

Usabilità e Utility sono due concetti diversi ma correlati e devono essere sempre buoni entrambi poiché importa poco se qualcosa è facile da usare se non è utile e non è buono se un sistema può ipoteticamente fare quallo che vuoi ma è difficile da usare. Quindi, per far sì che qualcosa sia utile, c'è bisogno che abbia una buona usabilità e utility. Quindi testare è essenziale per creare prodotti che incontrano le necessità degli utenti e le aspettative del principio dell'HCD. Come metodologia di ricerca dei problemi definiamo un usability-testing session in cui un ricercatore (chiamato facilitatore o moderatore) chiede ad un partecipante di completare delle attività utilizzando il prodotto o servizio in questione mentre gli altri ricercatori osservano e prendono appunti dai feedback.

Valutazione Euristica

Prima di testare con utenti reali, si può fare altro per scoprire i problemi di usabilità. Un metodo usato per valutare le interfacce utente è la valutazione Euristica che si basa su principi stabiliti dell'usabilità. Questa valutazione tipicamente richiede un piccolo team di valutatori, poiché può essere non facile completarla per un singolo individuo e comunque non essere molto preciso. Sebbene la valutazione euristica non fornisca un approccio sistematico per risolvere i problemi di usabilità o valutare le riprogettazioni, può guidare la creazione di progetti rivisti basati sui principi violati di buoni sistemi interattivi in modo efficiente.

I 10 principi di usabilità di Nielsen:

1. Visibilità dello stato del sistema : i designer dovrebbero tenere informati gli utenti su cosa sta succedendo attraverso feedback appropriati entro un tempo ragionevole.
2. Corrispondenza tra il sistema e il mondo reale : il design dovrebbe parlare il linguaggio degli utenti, con parole, frasi e concetti familiari all'utente, piuttosto che termini orientati al sistema.
3. Controllo e libertà dell'utente : gli utenti spesso scelgono le funzioni per errore e hanno bisogno di un "uscita di sicurezza" per uscire dallo stato indesiderato senza dover passare attraverso una lunga sequenza di passi.
4. Consistenza e standard : gli utenti non dovrebbero dover chiedersi se diverse parole, situazioni o azioni significano la stessa cosa, quindi il sistema deve avere una consistenza per tutte le funzioni.
5. Prevenzione degli errori : meglio che fornire un buon messaggio di errore, è meglio progettare un sistema in modo che gli errori non possano verificarsi in primo luogo.
6. Riconoscimento anziché ricordo : minmizzare il carico di cose da memorizzare per l'utente, rendendo gli oggetti, azioni e opzioni visibili.
7. Flessibilità e efficienza di utilizzo : progettare shortcut per gli utenti esperti, consentendo loro di velocizzare le interazioni più frequenti.
8. Design estetico e minimalista : l'interfacce non dovrebbero contenere informazioni irrilevanti ma solo quelle necessarie, poiché ogni elemento aggiunto aumenta la complessità del sistema.
9. Aiuto agli utenti a riconoscere, diagnosticare e recuperare gli errori : ogni messaggio di errore dovrebbe essere espresso in un linguaggio chiaro e semplice, indicando il problema e suggerendo una soluzione costruttiva.
10. Aiuto e documentazione : un buon design non ha bisogno di altre spiegazioni, ma è sempre una buona idea fornire documentazione o aiuto.

Gli elementi per un usability testing

- Facilitatore : guida i partecipanti nella sessione di test, spiega cosa sta succedendo e cosa si aspetta dall'utente, fa domande e prende appunti. Si assicura che i test siano di alta qualità e i dati validi.
- Tasks : sono attività realistiche che i partecipanti potrebbero fare nella vita reale. Possono essere molto specifici in base alla ricerca che vogliamo fare. Possono essere sia consegnati verbalmente che scritti ma ciò che è importante è la formulazione che deve essere capita dall'utente.
- Partecipanti : sono le persone che partecipano al test, idealmente dovrebbero essere selezionate a partire dalle personas che abbiaamo creato. Spesso gli viene chiesto dai facilitatori di pensare ad alta voce mentre completano i task in modo da capire quali sono gli step mentali per completare i task.

Ci sono 2 tipi di usability testing in base ai dati che collezzioniamo

- Dati qualitativi : costituiti da risultati osservativi che identificano le caratteristiche del progetto facili o difficili da usare.
- Dati quantitativi : sotto forma di una o più metriche, indicano se le attività erano facili da eseguire.

I test qualitativi tipicamente comportano la raccolta di dati attraverso domande a risposta aperta, osservazioni e feedback per ottenere informazioni di come gli utenti interagiscono con un prodotto. Spesso il test viene svolto negli stage iniziali del progetto per capire le necessità degli utenti, gli obiettivi e i punti deboli.

I test quantitativi è un metodo di ricerca che prevede la raccolta e analisi di dati numerici per misurare il comportamento e l'atteggiamento degli utenti nei confronti di un prodotto o servizio. In genere comporta la raccolta di dati tramite sondaggi, analisi e test a risposta chiusa. L'obiettivo è quello di dare ai designer dei dati statisticamente significanti che possono essere utilizzati per prendere decisioni. Possono essere utilizzate per valutare le decisioni che sono state prese durante il processo di progettazione.

Entrambi i testing sono importanti metodi nell'UX design e si completano l'un l'altro. I test qualitativi sono utili per capire il perché di un problema, mentre i test quantitativi sono utili per capire quanto è grande il problema.

Per pianificare un usability test abbiamo di:

- Definire gli obiettivi : cosa vogliamo scoprire? quali sono le nostre ipotesi?
- Definire le modalità : decidere la location, moderarli o meno e se farli di persona o in remoto
- Decidere numero di partecipanti : quanti partecipanti sono necessari per ottenere risultati significativi? (Nielsen indicava 5 per studi qualitativi e 20 per studi quantitativi)
- Scegliere i partecipanti giusti : per avere i risultati piu significativi possibili, i partecipanti devono rispettare il target d'utenza del prodotto
- Studiare i giusti task: i task devono essere realistici e pertinenti per il prodotto. Possono essere a domanda aperta per capire come gli utenti ragionano o specifici per capire se gli utenti riescono a completare un'azione
- Provare il test : prima di farlo con i partecipanti, è importante provare il test per capire se i task sono chiari e se il test è fattibile
- Decidere le metriche : i test qualitativi non hanno metriche poiché essendo fatti su poche persone non rappresentano l'intera popolazione mentre i test quantitativi hanno metriche ben definite per misurare l'usabilità.
- Scrivere il piano : una volta che abbiamo tutte le informazioni, è importante scrivere un documento per comunicare e registrare gli studi. Il documento dovrebbe contenere:
    - Nome del prodotto
    - Obiettivi dello studio
    - Logistica
    - Profili dei partecipanti
    - Tasks
    - Metriche e questionari

    - Descrizione del sistema

- Non fare tutto da solo : gli studi di usabilità possono essere utili a tutti i membri del team di progettazione, quindi è importante coinvolgere tutti i membri del team per avere più punti di vista e più idee.

10 Cose da evitare mentre si testa:

1. Dire agli utenti dove andare
2. Dire agli utenti cosa fare
3. Non aggiornare i dati dei task
4. Fare task troppo semplici
5. Creare uno scenario troppo elaborato
6. Scrivere task usando linguaggio di marketing o tecnico
7. Rischiare reazioni emotive
8. Cercare di essere divertente
9. Offendere i partecipanti
10. Fare altre domande oltre a quelle dei task