

Laboratorio di Human Computer Interaction- 2006

Appunti

Indice generale

| | |
|--|----|
| Laboratorio di Human Computer Interaction- 2006 | 1 |
| Appunti | 1 |
| 1. Introduzione | 1 |
| 2. Principi di usabilità | 2 |
| 2.1 Euristiche di Nielsen | 2 |
| 3. Bisogni & Requisiti | 3 |
| 3.1 Gli obiettivi del requirements management | 3 |
| 3.2 Come ricavare i requisiti | 3 |
| 3.3 Tecniche usate nella fase di definizione dei requisiti | 4 |
| 3.3.1 Questionari & Interviste : reclutamento | 4 |
| 3.3.2 Questionari & Interviste: caratteristiche | 5 |
| 3.4 Analisi | 5 |
| 3.5 Specifica – Approccio Goal Based | 5 |
| 3.5.1 AWARE..... | 5 |
| 3.5.2 Metodi matriciali goal based | 6 |
| 4. User Testing e Valutazione | 7 |
| 4.1 Metodi di Empirical Testing | 7 |
| 4.1.1 User Testing | 7 |
| 4.1.2 Contextual Analysis | 8 |
| 4.1.3 Focus Group – QUIS..... | 8 |
| 4.2 Metodi di Inspection | 8 |
| 4.2.1 Valutazione Euristica | 8 |
| 4.2.2 Cognitive Walkthrough | 9 |
| 4.2.3 Systematic Usability Evaluation (SUE)..... | 9 |
| 4.2.4 MILE..... | 9 |
| 4.3 Vantaggi e Svantaggi | 9 |
| 5. Design | 9 |
| 5.1 Scenari | 9 |
| 5.2 Story Board | 10 |
| 6. Prototipazione | 10 |

1. Introduzione

Interaction Design si occupa della progettazione di prodotti interattivi per fornire supporto alle persone nella loro vita quotidiana e professionale

Il processo di ID differisce in vari aspetti dal processo di sviluppo tradizionale dell'ingegneria del software:

- I **requisiti** sono incentrati sui bisogni dell'utente
- Il **design** si concentra sulle proprietà del sistema direttamente percepibili

dall'utente ed è inoltre auspicabile che l'utente stesso venga coinvolto nell'attività di design. C'è grande enfasi sulla prototipizzazione dell'interfaccia.

- La **valutazione** (di requisiti e design) coinvolge l'utente e si focalizza sulla soddisfacibilità dei suoi bisogni e sull'usabilità dell'interfaccia.

La **qualità** di un'applicazione dipende da parecchi fattori

1. Aderenza ai requisiti
2. Livello di partecipazione emotiva dell'utente (*engagement*)
3. Benefici ottenibili dall'utente e dal committente usando quella particolare applicazione
4. Usabilità: è l'efficacia, l'efficienza e la soddisfazione con cui specifici utenti possono conseguire specifici risultati in particolari contesti

2. Principi di usabilità

Si tratta di astrazioni di validità generale volte a orientare il design prendendo in considerazione i diversi aspetti.

- Rendere le cose visibili: più le funzioni sono visibili, più l'utente avrà chiaro cosa fare
- Fornire feedback: quando l'utente dà un comando, si aspetta di ricevere un riscontro in un tempo adeguato, quindi rendere la successiva possibilità d'interazione visibile all'utente
- Fornire vincoli: esigenza limitare le possibilità d'interazione tra utente e sistema. Il vantaggio è impedire all'utente di scegliere opzioni inappropriate riducendo quindi la possibilità di generare errori
- Fornire un mapping naturale: relazione tra i dispositivi di controllo e i loro effetti nel mondo. Rispettare le convenzioni
- Garantire la consistenza: progettare interfacce che usino operazioni simili, mediate da elementi simili (adottando regole chiare). Questo le rende più facili da apprendere.
- Fornire affordance: proprietà di un oggetto di far capire come va usato.

2.1 Euristiche di Nielsen

Valutazioni euristiche da interpretarsi secondo il contesto di design in cui vengono impiegate sulla base delle esperienze maturate.

1. Rendere visibile lo stato del sistema: mantenere sempre gli utenti informati di quello che sta succedendo, fornendo feedback adeguati in tempi ragionevoli
2. Matching tra il sistema ed il mondo reale: parlare il linguaggio dell'utente usando parole e frasi e concetti a lui familiari piuttosto che termini appartenenti al linguaggio del sistema
3. Abilitare il controllo dell'utente e lasciargli libertà: fornire sempre vie

d'uscita per permettere all'utente di abbandonare punti in cui si dovesse trovare inaspettatamente

4. Essere consistenti e seguire gli standard : evitare che l'utente debba chiedersi se parole, situazioni o azioni diverse significhino la stessa cosa
5. Aiutare l'utente a riconoscere, diagnosticare e correggere gli errori : usare un linguaggio semplice per descrivere la natura del problema e suggerire un modo per risolverlo
6. Prevenire gli errori : dove possibile fare in modo che gli errori non accadano
7. Aiutare il riconoscimento (ricognizione) piuttosto che il ricordo (recupero) : rendere oggetti, azioni ed opzioni ben visibili
8. Garantire flessibilità ed efficienza d'uso : fornire scorciatoie invisibili agli utenti inesperti ma che possono sveltire il lavoro di utenti esperti
9. Adottare un design accattivante e minimalista : evitare di dare informazione irrilevante o non necessaria
10. Fornire aiuti ed istruzioni documentali : se non se ne può fare a meno, fornire informazioni che possono essere consultate con facilità o che forniscano aiuto tramite una sequenza di passi che possano essere seguiti senza problemi.

3. Bisogni & Requisiti

Un requisito è un'affermazione relativa al prodotto che identifica una capacità, una caratteristica od un fattore di qualità che un sistema deve possedere per avere valore ed utilità per un utente.

Nel' Ingegneria del Software si hanno tradizionalmente requisiti funzionali (cosa deve fare il sistema) e non funzionali (proprietà e vincoli del sistema). Nell' Interaction Design richiede una comprensione delle funzionalità richieste e dei vincoli che limitano il funzionamento o lo sviluppo del prodotto , inoltre effettua una ulteriore categorizzazione più specifica dei requisiti non funzionali:

- Requisiti riguardanti i dati
- Requisiti ambientali / contestuali
- Requisiti utente
- Requisiti di usabilità

Questo perchè nell'IC la prospettiva è centrata sull'utente ed il focus è sull'interfaccia.

3.1 Gli obiettivi del requirements management

1. Definire i bisogni degli stakeholders e le situazioni d'uso
2. Definire i vincoli di progetto
3. Dire cosa il sistema deve o non deve fare
4. Tenere traccia delle decisioni prese durante la specifica del sistema

3.2 Come ricavare i requisiti

Esistono varie tecniche per esplicitare i requisiti di un'applicazione, in particolare si segue un processo iterativo: Elicitazione, Analisi, Specifica e Validazione. La validazione comporta feedback anche sulle fasi di Elicitazione ed Analisi.

- **Elicitazione:** scoperta di bisogni, caratteristiche e aspettative degli stakeholders. Effettuata tramite la raccolta di dati quantitativi e qualitativi.
- **Analisi:** discussione e riorganizzazione delle informazioni scoperte nella fase precedente. Assegnazione della priorità e soluzione di eventuali conflitti.
- **Specifica:** modellizzazione dei bisogni che il sistema deve soddisfare e delle proprietà percepibili dall'utente che il sistema deve avere
- **Validazione:** negoziare con gli stakeholders il consenso sugli obiettivi e sui requisiti

3.3 Tecniche usate nella fase di definizione dei requisiti

E' molto utile utilizzare queste tecniche in combinazione tra loro perchè si ottiene un'informazione più accurata e rappresentativa.

- Questionari: serve a rispondere a domande specifiche. Raccolgo dati qualitativi e quantitativi, posso raggiungere molte persone con poche risorse, non necessita di iterazione. Svantaggi: la progettazione è fondamentale, tasso di risposta può essere basso, risposte indesiderate, focus su questioni specifiche, no flessibilità
- Interviste: serve a esplorare determinate questioni. Raccolgo alcuni dati quantitativi ma prevalentemente dati qualitativi. L'intervistatore può guidare l'intervistato in caso di necessità, incoraggiando un contatto tra sviluppatori ed utenti, centrata sull'utente. Svantaggi: richiedono tempo, un ambiente artificiale potrebbero intimidire gli intervistati, richiedono rielaborazione dei dati qualitativi.
- Focus group & Workshop: gruppo di persone che fanno *brainstorming* con moderatore. Serve a raccogliere diversi punti di vista. Raccolgo prevalentemente dati qualitativi. Sottolineano le aree di consenso e di conflitto, incoraggiano il contatto tra sviluppatori ed utenti. Svantaggi: possono emergere figure dominanti.
- Osservazione sul campo (Contextual Inquiry): serve a comprendere il contesto dell'attività dell'utente. Raccolgo dati qualitativi. Permette di osservare il lavoro reale, ottenendo una visione che le altre tecniche non offrono. Svantaggi: richiede molto tempo e comporta una vastissima quantità di dati.
- Studio della documentazione: serve a imparare a conoscere procedure, regolamentazioni e standard. Raccolgo dati qualitativi. Non richiede impegno da parte degli utenti. Svantaggi: il lavoro di ogni giorno potrebbe essere diverso da quanto riportato nelle procedure.

3.3.1 Questionari & Interviste : reclutamento

Questionari ed interviste servono per avere un buon campione di utenti per la

raccolta dati. Tuttavia sono applicabili in ogni momento del processo di sviluppo del progetto (prima, durante e dopo)

1. **Distinzione** : “*today users*” vs “*would be users*”: clienti esistenti vs definizione di caratteristiche del potenziale nuovo mercato
2. **Selezione** : base iniziale di scelta. Evitare “prodotto per tutti”. Definire le caratteristiche del “target audience” (modello di utente finale) e profiling degli utenti.
3. **Filtro (Screening)**: scelta delle caratteristiche specifiche che gli oggetti dell'osservazione (profilo) e di altri soggetti che possono fornire informazioni utili (stakeholders), più individuazione di soggetti che rispondono a quelle caratteristiche.

Riguardo al campione, si distingue tra **Target Audience** (chi voglio raggiungere col prodotto), **Sampling Frame** (chi si riesce a raggiungere con il metodo di reclutamento) e **Sample** (chi si è effettivamente scelto). I problemi principali del reclutamento sono:

- Target Audience mal definito
- Sampling frame non rappresentativo o soggetto a preconcetti
- Sample non casuale, soggetto a preconcetti o di dimensione troppo ridotta (eccezione: regola dei 6 utenti di Nielsen)

3.3.2 Questionari & Interviste: caratteristiche

Metodologie basate su domande agli utenti, l'intervista prevede solo domande aperte a diretto contatto con l'utente.

Struttura generale dell'intervista: **profilazione approssimativa, warm up** (motivare i partecipanti), **concetti generali, approfondimento, retrospettiva, wrap-up**.

Struttura generale del questionario: **profilazione generale, stesura delle domande** (con feedback), **test pilota con campione minimo, eventuale riscrittura domande, scelta metodi statistici da usare, reclutamento, esecuzione, analisi dei dati**

Principi per domande chiuse: specificità (1 argomento per volta), chiarezza e non ambiguità, basarsi su esempi, consistenza, relazionabilità con l'esperienza dell'utente, no a situazioni estreme, top-down (dal generale allo specifico).

3.4 Analisi

Riceve come input un mix destrutturato di bisogni, aspettative, scenari, esempi, obiettivi ... registrati in differenti forma. Da qui in poi si analizza cosa fare per giungere alla soluzione di design. L'**analisi** è un processo iterativo, flessibile e revisionabile (feedback), con forte interazione degli stakeholders. Bisogna essere in grado di risolvere i conflitti, i fraintendimenti e deve esplorare soluzioni alternative.

3.5 Specifica – Approccio Goal Based

Documentazione e rappresentazione dei risultati del processo di analisi. Input

effettivo della fase di Design del prodotto. Rappresenta uno strumento di comunicazione con obiettivi specifici ed un pubblico target. E'anche uno strumento di validazione. Si realizza comunicazione tra stakeholders e progettisti assicurando coerenza e coordinamento tra i gruppi di lavoro.

La **specifica** dei requisiti deve specificare i requisiti cruciali tralasciando i dettagli, essere leggibile, tenendo conto delle assunzioni ovvie (ex. Usabilità), facile da capire e utile per progettisti e stakeholders (strutturata ma informale).

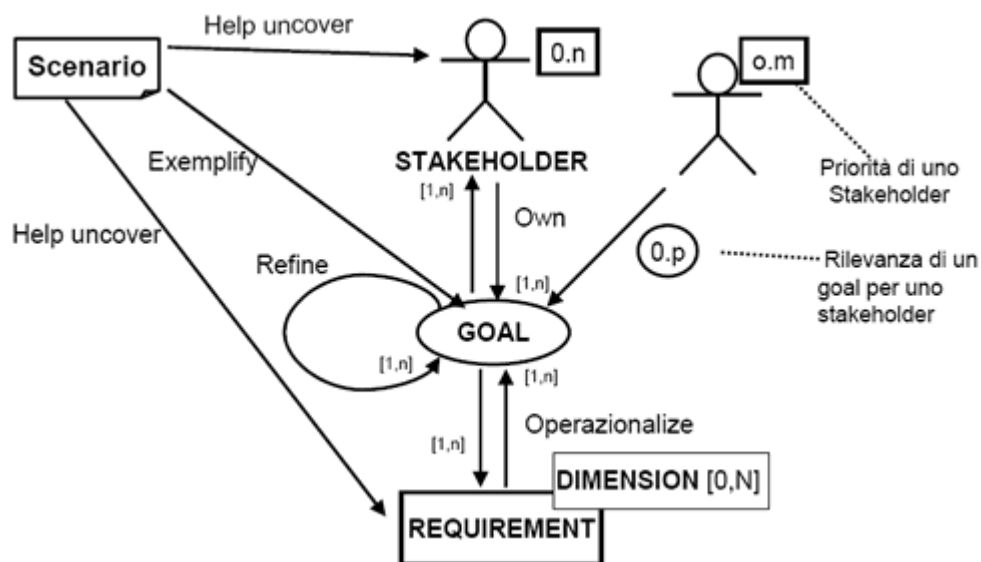
Nella specifica di tipo **goal based** si parte dai goal espressi dagli stakeholders per giungere ai requisiti, introducendo informazioni relative ai vincoli, agli scenari utente e alle risorse di tempo e di budget.

Si passa dallo spazio del problema (requisiti) allo spazio delle soluzioni (design). L'analista decide come risolvere obiettivi e bisogni degli stakeholder. Si formalizza una serie di **goal** per ogni stakeholder. Si scompongono i goal in subgoal ed infine in requisiti applicativi, tenendo traccia dei cambiamenti di obiettivo e delle decisioni ragionate prese sui requisiti.

Vantaggi: mette in relazione i requisiti con gli obiettivi che essi soddisfano, mettere in relazione i requisiti con il contesto dell'azienda e con il contesto di business, tener traccia le decisioni di design, facilitare la validazione, gestire i cambiamenti, supportare il riuso.

3.5.1 AWARE

Utilizziamo un approccio semi- formale **goal based (AWARE): Analysis of Web Application Requirements**. Metodologia e notazione per l'analisi e la specifica dei requisiti utente di applicazioni Web. Questo approccio aiuta ad evidenziare il “perchè” dei requisiti.



Tramite il processo di **raffinamento dei goal** si identificano quei goal che sono condivisi da più stakeholders: a quel punto tramite una scomposizione si suddivide il goal in subgoals in grado di rappresentare un'obiettivo più specifico,

fino a giungere alla definizione dei requisiti. Essi possono classificarsi secondo diverse dimensioni (che possono essere estese):

1. **Contenuto**
2. **Struttura del contenuto**
3. **Percorso di accesso**
4. **Navigazione**
5. **Presentazione**
6. **Operazione dell'Utente**
7. **Operazione del sistema**

I vantaggi di AWARE sono dati dalla registrazione dei risultati dell'elicitazione. Inoltre è uno strumento strutturato per l'analisi, utile a negoziare e comunicare con gli stakeholders. Aiuta a mantenere la tracciabilità, mostrando ai clienti che tutti i goal sono stati tenuti in considerazione e aiutando i progettisti a prendere decisioni basate sugli obiettivi reali dell'applicazione.

Tuttavia occorre fare attenzione: non prendere decisioni premature (che anticipino soluzioni di design), difficile bilanciare requisiti vaghi e decisioni troppo standard. Occorre guardare il problema da differenti prospettive per coglierne tutti gli aspetti fondamentali.

3.5.2 Metodi matriciali goal based

Sono metodi basati su notazioni a matrici multidimensionali. Realizzano relazioni N-M tra le seguenti dimensioni dei requisiti:

- Stakeholders
- Caratteristiche del profilo
- Goals
- Canali

4. User Testing e Valutazione

Esistono diversi metodi per valutare e quantificare l'usabilità di un'applicazione. Essi sono valutabili in diverse prospettive: oggetto, contesto, focus della valutazione, strumenti concettuali, partecipanti.

Principalmente, per quanto riguarda i partecipanti, si distingue in valutazione con utenti finali (metodi di **Empirical Testing**) e con esperti di usabilità (metodi di **Inspection**).

Gli strumenti concettuali inducono una nuova classificazione:

- **Task-based**: definizione di **scenari** e **task** con l'applicazione da validare
- **Model-based**: definizione di un modello (del comportamento ideale dell'utente – *metodo walktrough* - o dell'applicazione – *metodo SUE* -)
- **Heuristic-based**: definire un insieme di principi/attributi di usabilità di

natura euristica. Verifica dell'aderenza dell'applicazione a questi principi

Si può classificare a seconda del contesto

- **Usability Lab**
- **Contesto finale d'uso**
- **Contesto “informale”**

oppure seconda dell'oggetto della valutazione

- **Prototipo**
- **Specifica di design**
- **Prodotto finale**

Ovviamente è sempre possibile effettuare la valutazione di usabilità prendendo in considerazione vari punti di vista per formare un quadro completo più significativo

4.1 Metodi di Empirical Testing

Sono metodi di valutazione effettuati grazie ad un campione di utenti.

4.1.1 User Testing

E' un metodo **task-based**: utenti campione provano il sistema e sono osservati/controllati da esperti di usabilità che analizzano il loro comportamento e traggono conclusioni.

Come si prepara uno user test

1. Si reclutano gli utenti: quali e quanti utenti? Stesse considerazioni fatte per l'elicitazione
2. Si definiscono gli scenari d'uso e le priorità (tra quelli più rilevanti, per le features più utilizzate o pubblicizzate, o nelle aree sicuramente “problematiche”)
3. Gli esperti provano il test ed eventualmente lo aggiustano
4. Si definiscono i task da presentare all'utente (goal oriented, realistici)
5. Si definiscono i dati da raccogliere e la loro forma (qualitativi e quantitativi: dati aggregati, misurazioni singole, *success rate*, n° iterazioni, n° di cammini percorsi. Opinioni dell'utente: soddisfazione, engagement, facilità d'uso... si definiscono anche i metodi di rilevamento di questi dati)
6. Si preparano i materiali e l'ambiente di prova

Infine si effettua l'user test vero e proprio

- Si presenta all'utente uno o più scenari
- L'utente esegue un insieme di task assegnati (in media 5-6 per utente, da fare in un'ora al max – se no si stufa – lasciandolo a fare le cose da solo)
- Analisi delle osservazioni sull'esecuzione, con la partecipazione dell'utente

Chi valuta osserva (direttamente o indirettamente) gli utenti durante le sessioni d'uso. Il metodo si svolge in un contesto formale (usability lab) o semi-

organizzato. L'oggetto del metodo può essere un prototipo evolutivo o il prodotto finale.

I vantaggi sono dati dalla “freschezza” delle opinioni, dalla scoperta di problemi reali per l'utente, possono confermare o invalidare i risultati di altri metodi di ispezione.

4.1.2 Contextual Analysis

Si acquisiscono, tramite **osservazioni** ed **interviste** all'interno del **contesto d'uso** dei comportamenti e delle opinioni degli utenti dell'applicazione, generalmente eseguite su prototipi evolutivi oppure sul prodotto finale.

4.1.3 Focus Group – QUIS

FG: Derivata dalle ricerche di mercato. Si identificano problemi applicativi tramite discussioni di gruppo (effettuate a qualunque stadio del processo di sviluppo, quando necessario)

QUIS *Questionnaire for user interface satisfaction*: misura la percezione/soddisfazione dell'interfaccia da parte dell'utente tramite un questionario. Si effettua sul prodotto finale.

4.2 Metodi di Inspection

Uno (o più) esperti di usabilità (valutatori) **analizzano** in profondità l'applicazione e scoprono **errori di design** o potenziali **problemi d'uso** per l'utente. Gli strumenti concettuali, l'oggetto e il focus sono vari, il contesto è quello dove lavora l'esperto.

4.2.1 Valutazione Euristica

Si utilizzano linee guida (euristiche) rispetto alle quali valutare l'usabilità di un'applicazione. Normalmente si lavora in team. Si può effettuare ad ogni stadio dello sviluppo, in ogni contesto di prodotto.

Ci si rifà alle 10 Euristiche di Nielsen (che ha inventato questo metodo nel'91)

- Valutatori analizzano individualmente l'applicazione utilizzando le euristiche e fanno report al coordinatore
- Coordinatore crea elenco dei problemi rilevati
- Valutatori rivedono singolarmente le liste e assegnano punteggi di severità
- Coordinatore fa media dei punteggi e classifica in ordine di severità
- Progettisti cercano modelli e nuove soluzioni

E'utile impostare una metodologia comune tra i valutatori e il coordinatore (magari usando form standardizzati). Questa tecnica ha un ottimo rapporto costi/efficacia, e permette inoltre di standardizzare i risultati e di confrontarli con altre applicazioni che usano la stessa metodologia. I rischi sono la scoperta di problemi non rilevanti per l'utente finale, la soggettività e la necessità di disporre di esperti del settore.

4.2.2 Cognitive Walkthrough

Tipicamente usato in fase di design. Oggetto prototipo software oppure cartaceo. E' un metodo model- task based. Si modella e si simula il processo di problem-solving dell'utente e si valuta quanto questo sia naturale in merito all'uso dell'applicazione.

4.2.3 Systematic Usability Evaluation (SUE)

Utilizzato per valutare prodotti finali. Basato su euristiche, modelli e task. I modelli utilizzati sono generali ed indipendenti dal dominio applicativo, perciò richiedono al valutatore di farsi un modello preliminare dell'applicazione.

4.2.4 MILE

4.3 Vantaggi e Svantaggi

Vantaggi

- (ET) Buona valutazione di elementi percepibili dall'utente e della soddisfazione
- (INS) Efficace per identificare errori di design
- (INS) Costi "minori"

Svantaggi

- (ET) Utente osservato non è naturale (*Hawthorne effect*)
- (ET) Difficoltà di scelta utenti rappresentativi
- (ET) Costi elevati (Usability Lab, equipaggiamento)
- (INS) Servono persone di grande esperienza
- (INS) Può fornire analisi molto soggettive

La valutazione più completa si ottiene utilizzando sistematicamente entrambe le metodologie.

5. Design

Design: rappresenta lo spazio della soluzione, rappresentazione delle caratteristiche di **contenuti**, **servizi**, **meccanismi d'interazione**, **layout**. Può avere vari livelli d'astrazione e deve esulare dai dettagli implementativi

5.1 Scenari

Uno **scenario** è una storia d'uso. E' il racconto di come un utente interagisce con il sistema per raggiungere un determinato obiettivo

(profilo utente + obiettivo + contesto d'uso + tasks)

5.2 Story Board

Rappresentazione visiva realistica di come appare il sistema finale, priva di ogni

funzionalità. Si rendono le interazioni tramite annotazioni, vignette e schizzi. Schermate e Contesti d'uso

6. Prototipazione

Il **prototipo** è un artefatto che simula alcune (non tutte) proprietà d'interazione del sistema che si sta sviluppando

I prototipi si suddividono in tre tipi a seconda del loro scopo:

1. **Throw Away**: tecnologia fittizia, rende le idee per comprendere meglio il problema in fase iniziale di progetto
2. **Incrementale**: tecnologia definitiva, approccio modulare. Si integrano progressivamente tutti i componenti del sistema fino al completamento.
3. **Evolutivo**: tecnologia definitiva, copre tutti gli aspetti principali del sistema per quanto riguarda l'interfaccia ma trascura gli aspetti più legati al layer d'implementazione (performances, come effettivamente vengono svolte le operazioni)