Capitolo 5

Interfacce A Manipolazione Diretta

5.1. Introduzione
5.2. Linguaggio Grafico
5.3. Linguaggio Naturale

5.1. Introduzione

In ogni forma di dialogo valgono delle 'regole sociali' che bisogna seguire per evitare incomprensioni: rispetto di un linguaggio chiaro e condiviso che dipende dal dominio e dal contesto in cui il dialogo si svolge ma anche di regole di buona educazione. Nel caso dell'interazione con la tecnologia, si è pensato a lungo che l'unico vincolo da rispettare, nella definizione del linguaggio d'interazione, fosse la chiarezza: il progetto di una interfaccia usabile si è concentrato quindi sulla definizione di linguaggi che rispettassero questo vincolo. Questo è ancora vero, e vedremo come questi linguaggi possano essere definiti. Si è scoperto però più di recente che, nell'interazione con la tecnologia, valgono le stesse regole sociali che sono state studiate e vengono applicate nell'interazione fra umani: effetto attrazione/repulsione a seconda delle affinità o differenze nella personalità, attribuzione alla tecnologia di 'colpe' per errori commessi, fino alle reazioni emotive (da parte dell'utente) come il senso di frustrazione, la rabbia, l'orgoglio del successo, eccetera (vedi Scheda 5.1). Questa scoperta ha introdotto problemi nuovi nella progettazione dei linguaggi d'interazione, che sono meno noti e che analizzeremo in modo non approfondito alla fine di questo Capitolo.

Esaminiamo dunque un dialogo d'interazione fra utente e sistema: quali mosse esegue l'utente? Quali mosse esegue il sistema? Evidentemente la risposta è funzione dell'applicazione. Ma si può dire, in generale, che, se il ruolo del sistema è quello di *eseguire un compito (più o meno complesso) attivato da un obiettivo o da una richiesta esplicita dell'utente,* i ruoli dei due interlocutori saranno definibili nel modo seguente:

a. il sistema, nelle sue 'mosse', informa gli utenti su cosa possono fare in ogni fase del dialogo, sul suo stato interno (stadio di esecuzione dell'applicazione) e sui risultati intermedi o finali dei processi di elaborazione attivati (comprese le situazioni di 'errore'); può inoltre chiedere all'utente le informazioni che gli sono necessarie per eseguire un task;

b. l'utente, nelle sue 'mosse', sceglie cosa fare quando esistono diverse alternative possibili, risponde alle richieste d'informazione del sistema, chiede di annullare un comando precedente, manifesta al sistema il suo accordo su un suggerimento ricevuto.

E' possibile definire *stili d'interazione* che configurano combinazioni diverse dei tipi di mosse che abbiamo definito. Il primo fattore che determina lo stile è la definizione di *chi ha (in prevalenza) l'iniziativa* nel dialogo. Se è il sistema ad avere l'iniziativa (interazione *system-driven*), il dialogo assumerà, in prevalenza, una

forma in cui il sistema stesso fa domande o suggerisce azioni e l'utente risponde o esegue: esempio tipico, quello delle biglietterie automatiche (o *kiosk*) che si trovano nella maggior parte delle Stazioni Ferroviarie. Se invece è l'utente ad avere l'iniziativa (interazione *user-driven*), sarà lui a chiedere di eseguire un task e il sistema risponderà eseguendolo.

Scheda 5.1: Computers as Social Actors (Nass e coll)

Il Department of Communication dell'Università di Stanford conduce, da quasi dieci anni, studi orientati a dimostrare che 'le regole che guidano l'interazione fra esseri umani si applicano anche all'interazione con i calcolatori'. Qualche esempio:

- Regole di *buona educazione* (politeness): il comportamento di un calcolatore è visto come 'più corretto' se la richiesta di valutarlo viene dalla stessa macchina piuttosto che da un'altra;
- Effetto *genere*: i calcolatori con uscita vocale maschile sono visti come più adatti per il supporto alle decisioni mentre quelli con uscita vocale femminile sono visti come più adatti all'insegnamento e alla conversazione;
- Effetto *apprezzamento*: un calcolatore che elogia se stesso o critica altri calcolatori è visto come 'meno amichevole' di un calcolatore che elogia gli altri e critica se stesso; i calcolatori che criticano sono visti come 'più intelligenti' di quelli che elogiano;
- Effetto *gruppo*: quando gli utenti sono definiti come 'parte di un gruppo che comprende anche il calcolatore', tendono a dare punteggi maggiori alle sue prestazioni, ad essere più cooperativi ed a percepire il calcolatore come più simile a loro stessi;
- Effetto *somiglianza*: gli utenti che lavorano con un calcolatore che ha una 'personalità' simile alla loro tendono a incolparlo di meno di eventuali errori e a riconoscergli di più il merito di successi, rispetto a quelli che lavorano con una macchina con personalità diversa.

Il fattore di personalità rilevante nel creare questa differenza è la cosiddetta *dominanza*, cioè la tendenza a prendere decisioni o a comandare sull'altro. Nell'assegnare una personalità 'dominante' o 'sottomessa' al calcolatore, il primo aspetto manipolato negli esperimenti condotti era la *formulazione delle frasi in linguaggio naturale*: 'comandi' nel caso dominante (ad es: *"You should definitely rate the flashlight higher"*), 'suggerimenti' nel caso sottomesso (es: *"Perhaps the flashlight should be rated higher?"*). Altri aspetti rilevanti erano *l'ordine delle masse di dialogo* (il calcolatore 'dominante' parlava sempre per primo, il sottomesso faceva parlare per primo l'utente), il *livello di confidenza* che il calcolatore auto-assegnava alle sue azioni (alto per il dominante, basso per il sottomesso) e perfino il *nome* del calcolatore (*"Max"* per il dominante, *"Linus"* per il sottomesso).

Questi studi sembrano dimostrare che sarebbe possibile assegnare al calcolatore una personalità anche usando un'interfaccia grafica tradizionale (senza gli 'Embodied Agent' che vedremo nel Capitolo 6).

Esempio 5.1

Consideriamo il caso di un sistema che fornisca informazioni sui mezzi di trasporto. Vediamo le due forme d'interazione:

System-driven	*User-driven*
S:Dove vuoi andare?	U: Vorrei andare da Bari a Parigi in aereo. Mi dici gli orari ?
U:a Parigi	S: Non c'è un volo diretto: devi passare per Roma. Gli orari fra cui scegliere sono questi:
S: da dove parti?	.....
U: da Bari	.....
S: che mezzo vuoi usare?	
U: l'aereo	
.....	

La larga maggioranza dei sistemi con interfaccia grafica sono user-driven, mentre i sistemi d'installazione del software sono esempi tipici di interazione system-driven. In generale, si può affermare che i sistemi system-driven sono adatti a situazioni in cui completare un task richiede l'esecuzione di una sequenza ben definita di azioni da parte dell'utente, o in cui il sistema abbia ricevuto una 'delega permanente ad eseguire il task' da parte dell'utente: ad esempio, controllare virus o scaricare messaggi di posta elettronica appena arrivano. I sistemi user-driven sono invece quelli in cui la sequenza di esecuzione non è rigidamente predefinita. Un'applicazione potrà comprendere task con interfaccia user-driven e task con interfaccia system-driven.

Indipendentemente da chi, dei due interlocutori, ha l'iniziativa, nelle interfacce a manipolazione diretta il linguaggio con cui ciascuno di essi si esprime sarà una combinazione opportuna di manipolazione di oggetti grafici e presentazione di messaggi in linguaggio naturale. La realizzazione di una interfaccia usabile (secondo la definizione che ne abbiamo dato nel Capitolo 2) si basa sulla traduzione dei risultati dell'analisi di utenti, task e oggetti (effettuata come suggerito nel Capitolo 1) in un linguaggio appropriato, grafico e/o naturale. Inoltre, la descrizione dell'interazione mediante un metodo formale (Grammatica, Rete di Petri o altro) non deve restare una fase isolata del processo di progettazione ma deve tradursi in scelte realizzative.

5.2. Linguaggio Grafico

Un linguaggio grafico stabilisce una corrispondenza biunivoca fra un insieme di *concetti* ed un insieme di *oggetti grafici*, che d'ora in poi chiameremo *control.* Nei linguaggi grafici utilizzati nelle interfacce, i concetti sono i task eseguiti dall'applicazione, gli oggetti a cui questi task si applicano, i risultati intermedi prodotti e lo stato del sistema. Le interfacce grafiche si chiamano anche 'a manipolazione diretta' perché sono basate sul criterio che l'utente invii al sistema i comandi di richiesta di attivazione di task 'manipolando', con l'aiuto del mouse, dei control che rappresentano, sull'interfaccia, i task realizzabili. La manipolazione consiste nell'applicare ai control degli *eventi* appartenenti a classi ben definite. Ad esempio: (i) la '*scelta'*, mediante un click del mouse su un control e (ii) *l'applicazione di una funzione ad un file*, mediante l'applicazione dell'evento 'drag and drop' all'icona-file e all'icona-funzione (come nella cancellazione e nella compressione di un file, in cui l'icona-funzione è rappresentata, rispettivamente, con un 'cestino' o con una 'icona di compressione').

L'aspetto dei control può essere variato in modo da visualizzare situazioni temporanee di *non eseguibilità* di task. I control possono contenere informazione multimediale di vario tipo (combinazioni più o meno complesse di immagini e testi).

Se il task dell'applicazione non è molto complesso (e quindi la sua decomposizione gerarchica contiene un numero limitato di subtask elementari), tutti i control che rappresentano questi subtask possono essere mostrati in una sola finestra.

Esempio 5.2

Come abbiamo visto nell'esempio 2.9, Paint utilizza una sola finestra principale, con apertura di menu e pop-up window. Questa finestra (figura 5.1) contiene: un menu di bottoni (che rappresenta i task di manipolazione del file-immagine) e due menu di icone, uno verticale (che rappresenta i task di 'costruzione' dell'immagine) e uno orizzontale (che rappresenta la tavolozza dei colori). Una barra di stato (in basso) completa la finestra.

I task associati alla toolbar di bottoni (e visualizzabili nei menu a tendina) sono rappresentabili nella seguente gerarchia:

File

- Nuovo
- Apri ➡F
- Salva
- Salva con nome ➡F
- ---
- Anteprima di Stampa ➡F
- Imposta pagina ➡F
- Stampa ➡F

Gestisci il file-immagine

Modifica

- Annulla
- Ripeti
- Taglia
- Copia
- Incolla
- Cancella area selezionata
- Seleziona tutto
- ---
- Copia su
- Incolla da

Gestisci parti dell'immagine

Visualizza

- Casella degli strumenti
- Tavolozza
- Barra di stato
- Barra degli strumenti di testo
- ---
- Zoom
- Visualizza bitmap
- **????**

Organizza la finestra di lavoro

- Immagine
  - Capovolgi/ruota
  - Allunga/inclina
  - Inverti colori
  - Attributi
  - Cancella immagine

Gestisci l'intera immagine

- Colori
  - modifica colori

Scegli i 'colori'

Confrontando questa gerarchia con quella rappresentata nella figura 2.8, si nota una sostanziale corrispondenza fra gerarchia dei task dell'applicazione e organizzazione dei control nell'interfaccia.

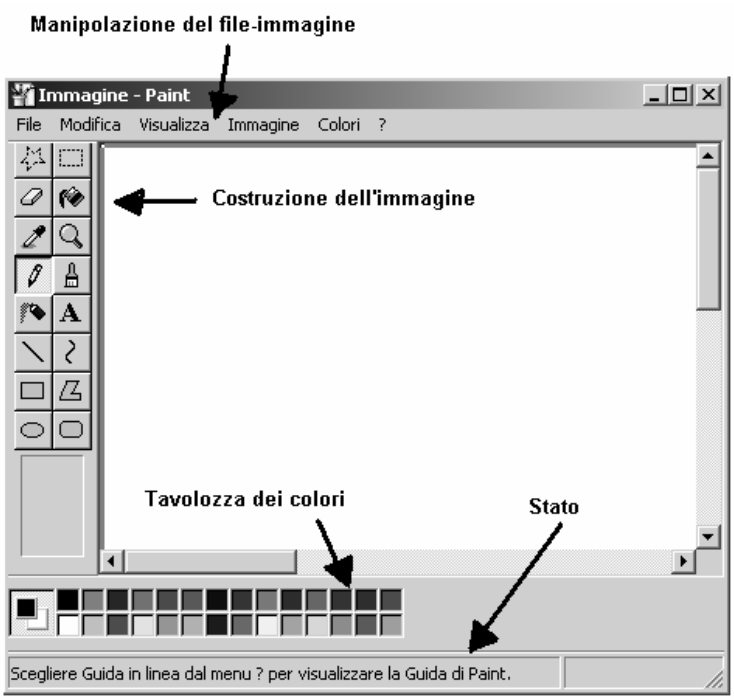

Figura 5.1: la finestra principale di Paint

Se invece il task è complesso, per ridurre la complessità visiva dell'interfaccia non tutti i subtask sono visualizzati nella finestra principale: alcuni di essi richiedono l'apertura di nuove finestre, sovrapposte o no alla precedente. I control che rappresentano lo stato del sistema vengono invece mostrati soltanto quando è necessario, in genere in modo temporaneo, in pop-up window (come nella figura 5.2).

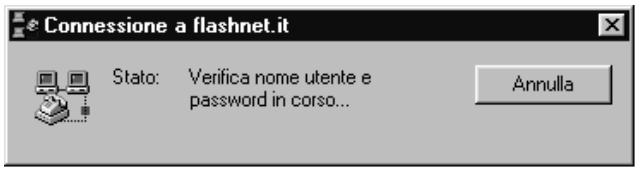

Figura 5.2: un esempio di messaggio sullo stato del sistema in Eudora

Display temporanei più complessi vengono utilizzati per richiedere all'utente informazioni sui modi di eseguire un task, come nella figura 5.3.

Tutto questo è ben noto a chiunque abbia usato un qualsiasi sistema a interfaccia grafica. Quello che è meno noto è perché alcuni di questi sistemi siano, di fatto, usabili ed altri no, e quali criteri occorra seguire per avere maggiori garanzie di usabilità.

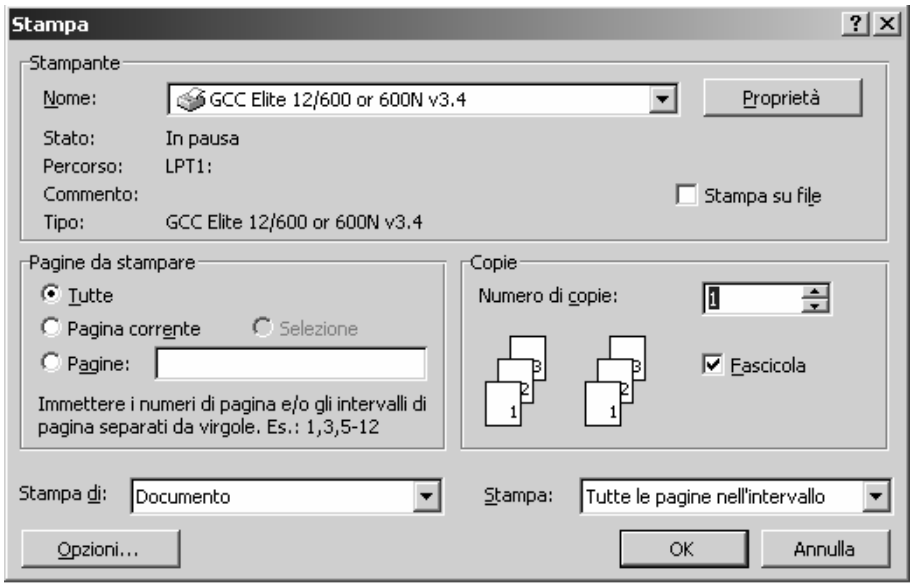

Figura 5.3: una finestra di stampa

a. raggruppamento dei task

Nel definire come raggruppare in una finestra, una toolbar e/o un menu a tendina i task della gerarchia, occorre tener conto della doppia esigenza che abbiamo discusso nel Capitolo 2: ridurre la *complessità* funzionale (soprattutto dei task più frequenti) contenendo nello stesso tempo la complessità visiva. Minimizzare la complessità funzionale porterebbe a rappresentare *tutti* i nodi-foglia della gerarchia dei task in un'unica finestra, come avviene in Paint: in questo modo, l'utente potrebbe accedere immediatamente a tutti i control che permettono di eseguire task elementari. Minimizzare la complessità visiva porterebbe invece a riprodurre, in ogni display, una minima parte della gerarchia dei task: al limite, un solo livello. In questo modo, i control rappresentati in ogni finestra sarebbero pochi, ma il numero di 'azioni' che l'utente dovrebbe compiere per eseguire un task complesso sarebbe molto maggiore. Il rischio di una soluzione di questo genere sarebbe inoltre quello del cosiddetto '*disorientamento cognitivo'*, secondo il quale, dopo l'apertura di un certo numero di finestre sovrapposte, l'utente non si rende conto più di 'dove si trova' e 'come tornare indietro', nello spazio delle finestre successivamente aperte dall'applicazione.

La soluzione di compromesso consiste nel rappresentare, in ogni finestra, un numero *ragionevole* di oggetti: questo numero dipende ovviamente dal formato del display e dalla modalità d'interazione.

I control sono raggruppati, nelle toolbar, secondo criteri di somiglianza. Si veda, ad esempio, la barra degli strumenti 'standard' in Word (figura 5.4), che contiene otto gruppi di icone che riproducono, in forma di immagine, alcune delle funzioni più frequenti rappresentate anche nei menu a tendina della barra di bottoni; i gruppi sono definiti per analogia. Ad esempio, il primo mette insieme le operazioni di 'nuovo', 'apri' e 'salva', applicati ad un file; il terzo, quelle di taglia, copia, incolla e copia formato, ecc. Il raggruppamento rispetta quello degli item nei menu a tendina.

Figura 5.4: raggruppamento delle icone nella barra degli strumenti 'standard' di Word.

Anche quando la suddivisione in gruppi non è indicata esplicitamente, le icone possono essere collocate nelle toolbar in modo da rispettare criteri di somiglianza. Si veda, ad esempio, la distribuzione dei control nella toolbar per la costruzione di immagini in Paint (figura 5.5).

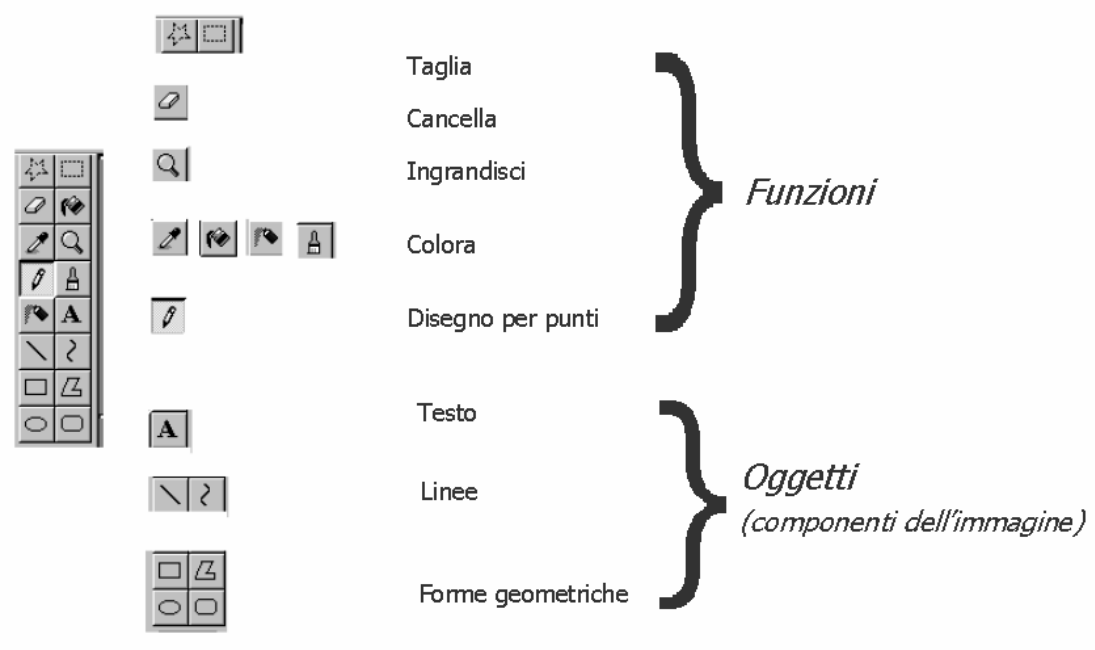

Figura 5.5: raggruppamento di oggetti in una toolbar di Paint

b. navigazione fra display

> ***Definizione 5.1***
>
> Chiamiamo *Grafo dei Display* (o delle finestre) un grafo in cui:
>
> ad ogni nodo è associato un nuovo display del sistema,
>
> ad ogni arco è associata una azione che permette di passare dal display 'padre' al display 'figlio',
>
> il nodo-radice rappresenta il 'display iniziale',
>
> i nodi-foglia rappresentano 'display terminali'.

Un *cammino* (sequenza di archi) che conduce dal nodo-radice ad un nodo-foglia rappresenta la sequenza di azioni che conducono dal display iniziale ad un display terminale, con i display intermedi mostrati in successione durante l'esecuzione di un task[1].

Il grafo dei display può assumere una delle strutture mostrate in figura 5.6. In una struttura ad albero, non esistono percorsi alternativi che portano allo stesso stato dell'interfaccia: questi percorsi esistono invece nei grafi aciclici. In entrambe queste strutture, il ritorno ad uno stato di livello superiore richiede una o più azioni di 'chiusura' di finestre. In un grafo con cicli, invece, è possibile, in alcune fasi, tornare con una sola azione a stati di livello superiore

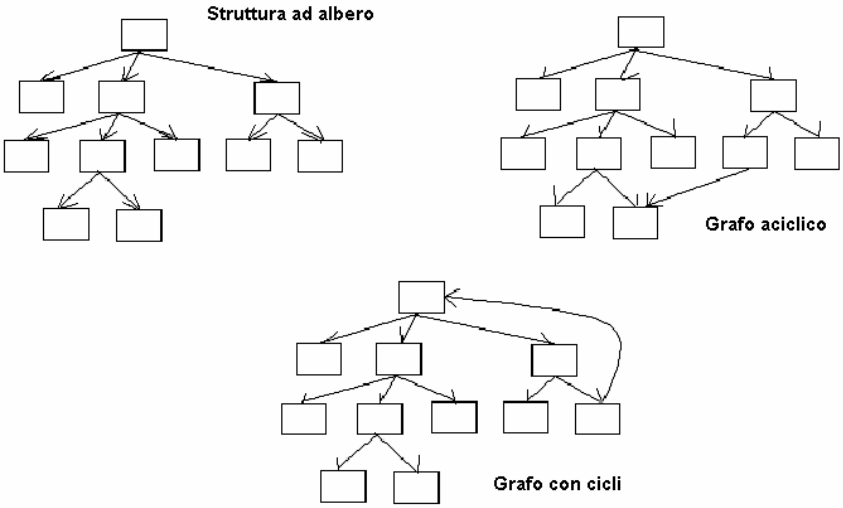

Figura 5.6: Strutture possibili per il Grafo dei Display

Esempio 5.5

L'apertura di finestre successive che specificano le proprietà di stampa in Word è rappresentabile in un grafo aciclico. L'operazione di Exit attraverso manipolazione di un control presente in ogni fase dell'interazione è invece rappresentabile con un ciclo che riporta al nodo radice.

c. rappresentazione dei task

Come abbiamo detto nel Capitolo 1, un task è descritto da una frase che specifica una *funzione* da applicare ad un *oggetto*, con eventuali *parametri.*

Esempio 5.6

In un text-editor:

crea un nuovo documento

apri un documento

[1] Nota la differenza rispetto ad una Rete di Petri: un display (o finestra) contiene diversi control. Quindi, un cammino in una RP è, in genere, molto più lungo di un cammino nel Grafo dei Display e rappresenta tutte le azioni che conducono dal display iniziale al display finale, in modo più dettagliato ed evidenziando (come abbiamo detto nel Capitolo 4) relazioni di scelta, concorrenza, indipendenza d'ordine, ecc.

salva un documento

stampa un documento

cancella il testo isolato

copia il testo isolato

……

In un image-editor:

colora la forma indicata

inserisci una forma quadrata

……

Una frase deve essere rappresentata in un control che può essere un bottone, un'icona, un item da menu testuale,... Poiché è difficile rappresentare in un solo simbolo o una sola stringa di piccola dimensione l'intera frase, si sceglie in genere di rappresentare, nel control, la funzione o l'oggetto, omettendo l'altro elemento: la scelta dell'elemento da omettere si basa sull'ipotesi che questo sia chiaro per l'utente, anche senza che venga specificato. Quando però si ipotizza che tutti gli elementi debbano essere rappresentati, si possono utilizzare control più complessi come le icone 'miste' (cioè composte da immagine + label).

c.1. rappresentazione testuale

I bottoni e gli item di menu contengono stringhe di lunghezza limitata (e quindi informazione limitata): nella loro definizione, viene quindi fatta una scelta precisa di 'cosa rappresentare' e 'cosa omettere', assumendo che sia implicitamente comprensibile.

Esempio 5.7

I bottoni del menu principale di Word (figura 5.7) rappresentano task nominando in alcuni casi l'oggetto (es 'file' o 'tabella'), in altri la funzione (es 'modifica' o 'inserisci'). Gli elementi dei menu a tendina specificano l'elemento mancante.

File
Nuovo : crea un nuovo documento
File
Apri : apri un documento
File
Salva : salva il documento
... ...

Modifica
Annulla : annulla l'ultimo comando
(qui, l'oggetto 'comando' non è specificato)
Modifica
Taglia : taglia la parte di testo isolata
(qui, l'oggetto 'testo' non è specificato)
Modifica
Copia : copia la parte di testo isolata
(qui, l'oggetto 'testo' non è specificato)
... ...

Figura 5.7: Formulazione di alcuni task nel menu principale di Word

c.2. rappresentazione mediante icone

Le icone rappresentano, con una immagine, task o stati del sistema. L'immagine non va scelta con criteri puramente estetici, ma deve rispettare criteri di comprensibilità.

Rispetto alla *funzione* svolta, le icone possono essere:

interattive	se consentono all'utente di inviare comandi all'applicazione
non interattive	se sono indicatori sullo stato dell'applicazione

Rispetto all'*oggetto denotato*, possono rappresentare:

funzioni	da svolgere (es: stampa, cancella)
oggetti	ai quali applicare le funzioni; questi possono essere 'concreti' (ad esempio, una stampante) o 'astratti' (es: un file, un paragrafo, una immagine,…)

Rispetto al *rapporto fra oggetto denotato e simbolo utilizzato* per denotarlo, possono adottare una rappresentazione:

per somiglianza	(se l'immagine riproduce l'oggetto in modo più o meno fedele)
per esempi	(se l'immagine riproduce un particolare elemento della categoria da rappresentare)
simboliche	(se viene utilizzato un simbolo universalmente considerato come significativo per denotare l'oggetto o la funzione)
arbitrarie	(se il simbolo utilizzato non è universalmente noto nella iconografia comune, ma lo è (o lo diventa) nell'ambito dell'HCI.

La figura 5.8 rappresenta una classificazione delle icone secondo i criteri appena definiti.

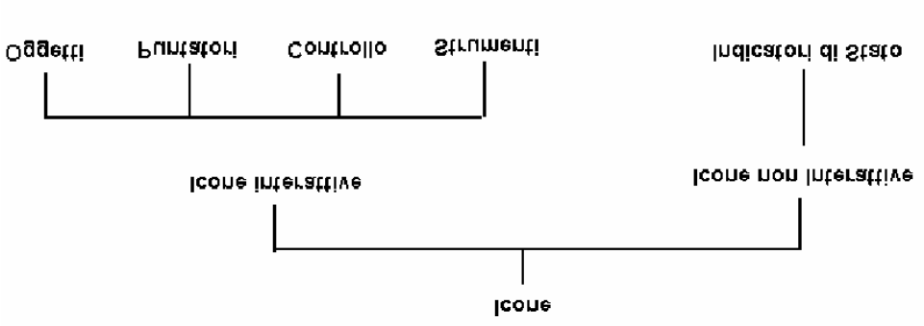

Figura 5.8: Una classificazione delle icone

Esempio 5.8

Le icone di Windows nella figura 5.9 denotano alcune funzioni con gli oggetti utilizzati per compierle: una gomma per 'cancellare', una penna per 'disegnare', un dischetto per 'salvare', ecc) utilizzando criteri di *somiglianza* per la rappresentazione di questi oggetti. In altri casi, viene omessa la funzione per rappresentare soltanto l'oggetto (il 'documento' da creare) o viene data ua rappresentazione dinamica della funzione ('apertura' di una cartella).

Figura 5.9: Oggetti e strumenti in Windows

Esempio 5.9

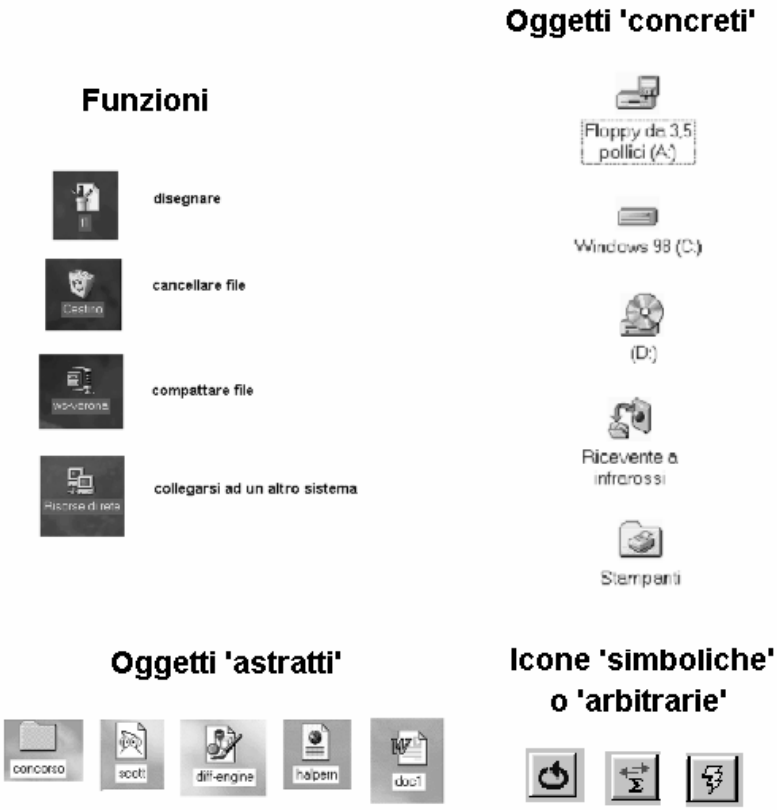

Figura 5.10: Esempi di icone interattive

La figura 5.10 contiene alcune rappresentazioni iconiche di funzioni in applicazioni in Windows. Gli oggetti concreti denotano parti di un calcolatore. Gli oggetti astratti denotano il formato del documento (word, gs, paint ecc), con simboli che rappresentano il nome del programma (un 'fantasma' come metafora in ghostview) o quello che il programma fa (scrivere su un foglio, per word). La prima icona simbolica è il simbolo ormai universale di 'undo', mentre le altre due sono simboli arbitrari per 'propagazione di evidenza in un belief network' e 'compilazione, con calcolo dei parametri, in un belief network'

Esercizio 5.1

Di quali delle icone in figura 5.11 conosci il significato? Di quali, se non lo conosci, puoi immaginarlo?

Figura 5.11: esempi di icone

Per completare l'informazione rappresentata in un'icona, è possibile visualizzare una label in fase di 'scorrimento' con il mouse o associare all'immagine una label all'interno della stessa icona. Nelle icone miste, la label può essere utilizzata per replicare la descrizione della funzione o per rappresentare l'oggetto a cui questa funzione si applica. Esempi tipici sono le icone di GS view e di WinZip (figura 5.12)

Figura 5.12: Esempi di icone miste

d. indicatori di stato

Le icone che rappresentano indicatori di stato sono mostrate, in genere, all'interno di pop-up window che contengono un messaggio in linguaggio naturale (per garantirne la chiarezza) e un bottone di conferma. La figura 5.2 mostrava un primo esempio di questo genere di pop-up window in cui l'icona di stato rappresentava, 'per somiglianza', il processo di trasferimento di dati fra due sistemi.

Esempio 5.10

La pop-up window in figura 5.13 contiene un messaggio dinamico e complesso che descrive l'evoluzione dello stato del sistema; l'icona di stato è rappresentata da una barra che si riempie progressivamente.

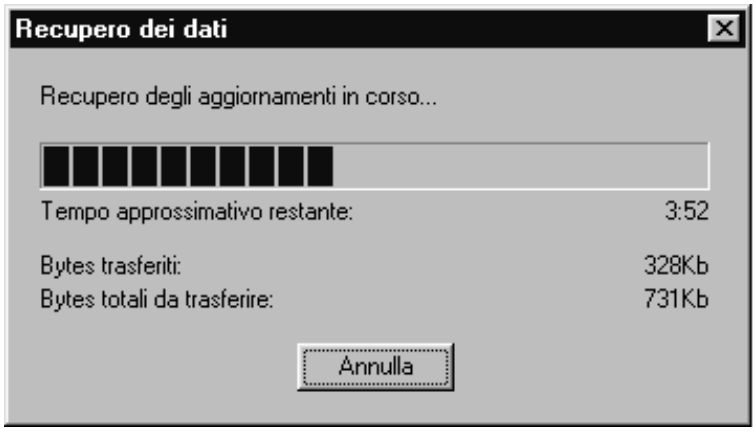

Figura 5.13: Una finestra di descrizione dinamica dello stato del sistema

e. dimensione, forma esterna e colore dei control

La dimensione dei control è funzione da un lato del numero degli oggetti che bisogna mostrare simultaneamente sul display, dall'altro delle condizioni in cui il sistema dovrà essere utilizzato. Nelle applicazioni di window su un pc, icone e bottoni sono di piccole dimensioni e le finestre di pop-up non oscurano completamente la finestra principale, per evitare il 'disorientamento cognitivo'. Invece, allo stato attuale della tecnologia, in un palmare l'interazione è meno agevole che con un mouse, per le ragioni che abbiamo descritto nel Capitolo 2 (uso in movimento, touch screen con selezione con piccola penna ottica, ecc). Quindi, gli oggetti grafici devono avere dimensioni più grandi. D'altra parte, anche la dimensione dello schermo è molto più ridotta. Ne consegue che il *numero di control* rappresentabili in un singolo display è limitato e che, per assicurare una complessità funzionale accettabile, i task devono essere semplificati (vedi un esempio nella Scheda 5.2). Un problema analogo di precisione nella selezione degli oggetti grafici si pone nei sistemi che utilizzano un touch-screen: qui, la dimensione dei control deve essere maggiore, a parità di dimensioni dello schermo utilizzabile.

Un commento sulla 'personalità' del sistema: diversi studi, del Gruppo dell'Università di Stanford citato nella Scheda 5.1 e all'IBM, hanno mostrato che un sistema può mostrare una personalità non solo e non necessariamente nei suoi messaggi in linguaggio naturale o basati su agenti animati, ma anche, semplicemente, mediante la scelta di forma e colore dei suoi oggetti grafici.

Esempio 5.11

Osserva la dimensione dei bottoni nelle biglietterie ferroviarie automatiche e commenta.

Forma esterna e colore possono essere utilizzati, nelle icone, per rappresentare parte dell'informazione che l'icona intende trasmettere. Una prima distinzione può essere utilizzata per differenziare 'icone interattive'

(forma esterna quadrata) da 'icone non interattive' (forme esterne diverse). Per queste ultime, possono essere utilizzati simboli di uso comune, come quelli impiegati nei segnali stradali, per denotare 'pericolo' o 'divieto'. Nota, ad esempio, i simboli utilizzati nelle finestre di messaggio mostrate nella Sezione 5.3. per denotare 'attenzione', errore' e 'informazione'.

Esercizio 5.2

Proponi icone per rappresentare i concetti seguenti:
Attenzione: c'è il pericolo di perdere dati introdotti.
Non puoi, uscire dall'applicazione senza prima salvare i dati
I dati che hai introdotto sono corretti
I dati che hai introdotto sono errati
Utilizza forma esterna e colore delle icone, oltre all'immagine rappresentata (ed eventualmente alla label di scorrimento). Giustifica le tue scelte.

f. consistenza interna ed esterna

In un'interfaccia consistente, gli oggetti grafici che rappresentano task uguali (o simili) sono collocati nello *stesso ordine* (o in ordine simile) sul display e all'interno delle toolbar e dei menu. Questo vincolo vale all'interno della stessa applicazione per la consistenza interna, e di applicazioni diverse nello stesso ambiente per la consistenza esterna.

Esempio 5.12

Confrontiamo l'ordine degli elementi nel menu a tendina collegato a 'File', in PowerPoint e in Paint: notiamo che gli elementi in comune alle due applicazioni sono presentati nello stesso ordine.

File
- Nuovo
- Apri ➨F
- Chiudi
- Salva
- Salva con nome ➨
- Salva come HTML
- Presentazione portatile
- ---
- Anteprima di Stampa ➨F
- Imposta pagina ➨F
- Stampa ➨F

File
- Nuovo
- Apri ➨F
- Salva
- Salva con nome ➨
- ---
- Anteprima di Stampa ➨F
- Imposta pagina ➨F
- Stampa ➨F

Esercizio 5.3

Confrontiamo (nella figura 5.14) tre applicazioni per la scrittura di testi in Windows: le tre applicazioni (word, wordpad e blocconote) sono via via più semplici e quindi hanno, in comune, un sottoinsieme di funzioni. Prova ad esaminare le toolbar e a riflettere sui criteri di consistenza esterna: individua casi di consistenza e casi di inconsistenza.

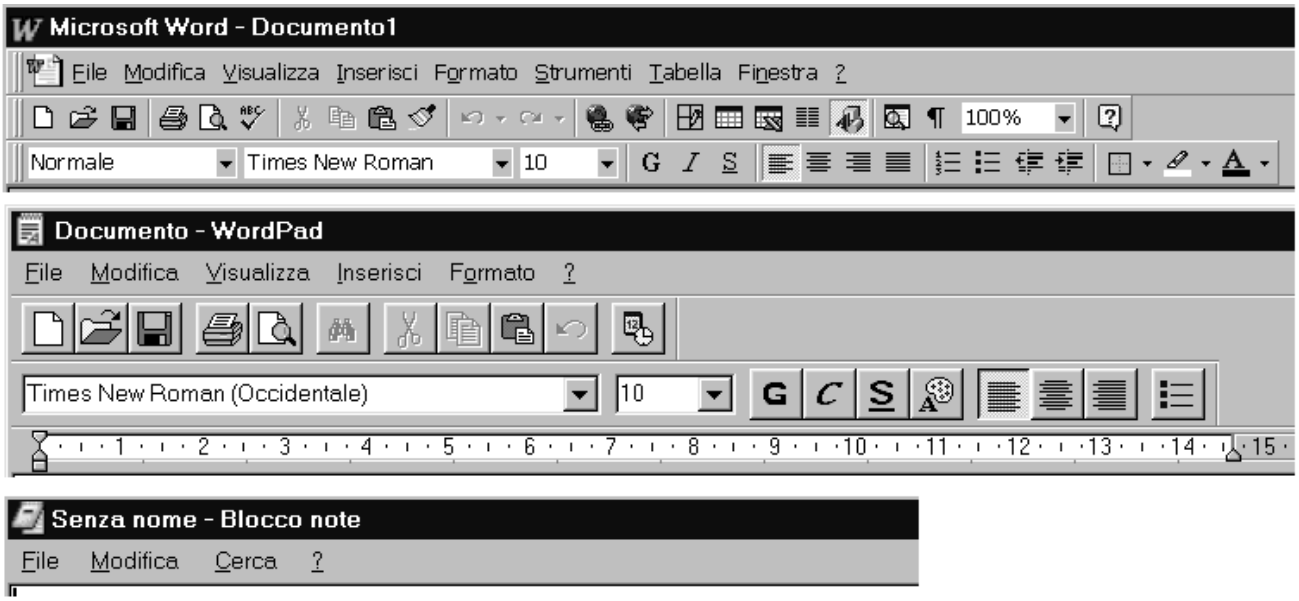

Figura 5.14: toolbar in programmi di scrittura di Office

Esercizio 5.4

Confrontiamo (nella figura 5.15) due sistemi di posta elettronica, entrambi all'interno di Windows: si riscontrano, qui, innumerevoli esempi di inconsistenza visiva esterna. Prova ad individuarne alcuni.

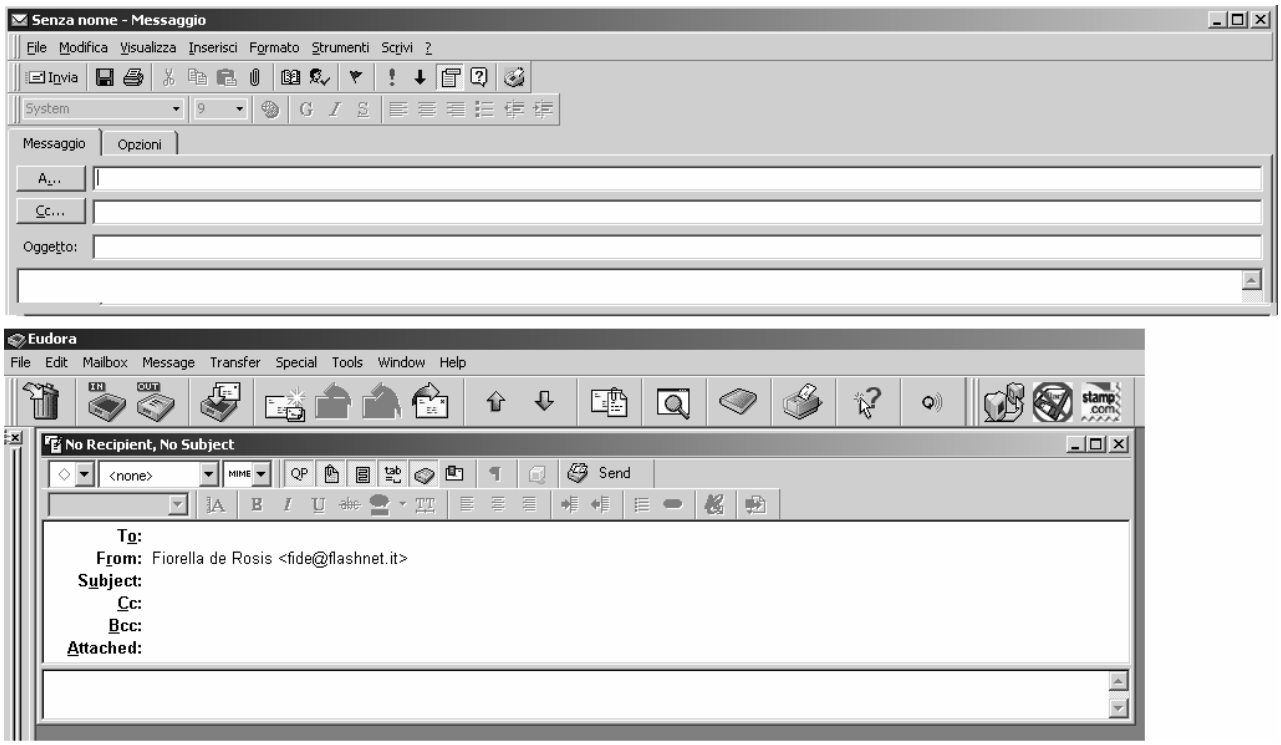

Figura 5.15: toolbar in due programmi di posta elettronica

Una volta stabilita l'organizzazione degli oggetti nei display, la definizione di un linguaggio grafico consistente contribuisce in modo determinante alla realizzazione della consistenza visiva. Il primo passo

nella creazione di questo linguaggio è nella scelta di una *metafora d'interazione*, da applicare per la definizione dei control. Nelle applicazioni di windows, le due metafore uniformemente applicate sono quella della 'scrivania' (per le scritture di testi) e del tavolo da disegno' (per la preparazione di disegni). In altre applicazioni (come Hugin) questa metafora è estesa con la rappresentazione di simboli specifici dell'applicazione (icone 'per somiglianza', per rappresentare nodi e archi nei grafi e icone 'simboliche' o 'arbitrarie' per rappresentare le funzioni di costruzione del grafo, compilazione, propagazione dell'evidenza, ecc). Una volta scelta la metafora, occorre definire la mappatura task-control con criteri di razionalità: *funzioni o oggetti simili devono essere rappresentati con simboli o stringhe simili.* Questo è semplice per i task elementari 'comuni' di cui abbiamo parlato nel Capitolo 1 (come il task di Exit o di Stampa), che devono essere rappresentati in modo identico; contrassegnare con una label i task comuni, nella gerarchia, aiuta a raggiungere questo obiettivo.

Esempio 5.13

La figura 5.16 mostra rappresentazioni iconiche consistenti delle funzioni di 'undo' (o 'annulla') e di 'elimina' (o 'taglia') in diverse applicazioni di windows.

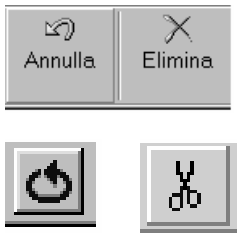

Figura 5.16: consistenza esterna nella rappresentazione di undo e taglia

Il problema diviene più complesso quando occorre rappresentare stesse funzioni applicate a oggetti diversi, o viceversa. Nei due casi, il simbolo (o la label) utilizzati per rappresentare la funzione o l'oggetto devono essere gli stessi. Vedremo degli esempi di costruzione di un linguaggio iconico consistente nel Capitolo 7, sul Caso di Studio considerato in queste Dispense.

Esercizio 5.5

Prova a rappresentare, con icone miste 'consistenti', le seguenti funzioni, in un sistema elettronico per l'ordinazione di pizze in un Ristorante:

compila un'ordinazione
invia un'ordinazione
modifica un'ordinazione
compila un messaggio per un altro tavolo
invia il messaggio
modifica il messaggio

Giustifica le tue scelte discutendo perché il linguaggio che hai definito è consistente.

g. personalizzazione

Come abbiamo discusso nel Capitolo 1, il primo livello di personalizzazione consiste nel *definire vincoli sui task eseguibili:* questo livello si rifletterà nella visualizzazione, sull'interfaccia, soltanto degli oggetti grafici che rappresentano questi task. [2] Se inoltre *l'ordine degli item nei menu* è funzione della frequenza con cui le funzioni associate vengono eseguite, anche quest'ordine potrà essere personalizzato. In alternativa, *le funzioni più frequenti* per la particolare categoria di utenti considerata potranno essere rappresentate in un menu separato. Un esempio classico in questo senso è dato da word, che permette di personalizzare la barra di formattazione.

Ma il livello più raffinato di personalizzazione è certamente quello che abbiamo introdotto nel Capitolo 3, in cui all'insieme delle funzioni e degli oggetti vengono associati simboli grafici diversi (scelti, ad esempio, a seconda della cultura, della lingua). Nel Capitolo 6 vedremo la forma più avanzata di personalizzazione, in cui l'intero contenuto di un messaggio (in linguaggio naturale o multimediale) viene adattato alle caratteristiche dell'utente.

Esercizio 5.6

Considera i task svolti da un 'Comunicatore di messaggi in un piccolo gruppo' e prova a definire icone personalizzate per bambini e adulti, per i task seguenti: calendario, scrittura messaggi, invio messaggi, disegno, rappresentazione degli utenti nel gruppo.

5.3. Linguaggio Naturale

I messaggi che il sistema mostra nelle finestre di pop-up non sono neutri ma manifestano una sua personalità. Quando segnalano un errore commesso dall'utente, possono farlo in modo più o meno 'empatico', per evitare di urtare la sua suscettibilità. Quando segnalano il risultato (negativo o positivo) di una elaborazione, possono farlo in modo da ridurre il senso di frustrazione o partecipare al senso di soddisfazione dell'utente. La finestra che contiene il messaggio può contenere anche un icona che simboleggia il 'tipo' di messaggio mostrato; in ogni caso, deve contenere un bottone 'di conferma' che ha, come scopo, di verificare che l'utente abbia recepito il messaggio

a. messaggi 'neutri' di 'attenzione' e 'errore'

La prima figura 5.17 mostra un messaggio di errore durante il processo di stampa di un documento; la seconda, un messaggio prodotto da Eudora quando l'utente dimentica di inserire indirizzo o nickname del destinatario in un email. Entrambi i messaggi hanno un tono 'neutro'. Notare però che, nel secondo, non si dice all'utente 'hai sbagliato: non hai inserito il nome del destinatario' ma si specifica 'come è fatto un messaggio corretto' e si suggerisce all'utente di controllare se non ha, per caso, commesso un errore

[2] Un'alternativa possibile consiste nel mostrare tutti i task, disattivando quelli 'non permessi': ma questo (si sostiene) rischierebbe di generare, nell'utente, in senso di frustrazione o di esclusione.

comune (un esempio di personalità *non dominante*). Nel messaggio 'di attenzione' (figura 5.18) si avverte l'utente delle possibili conseguenze negative dell'azione che sta per fare.

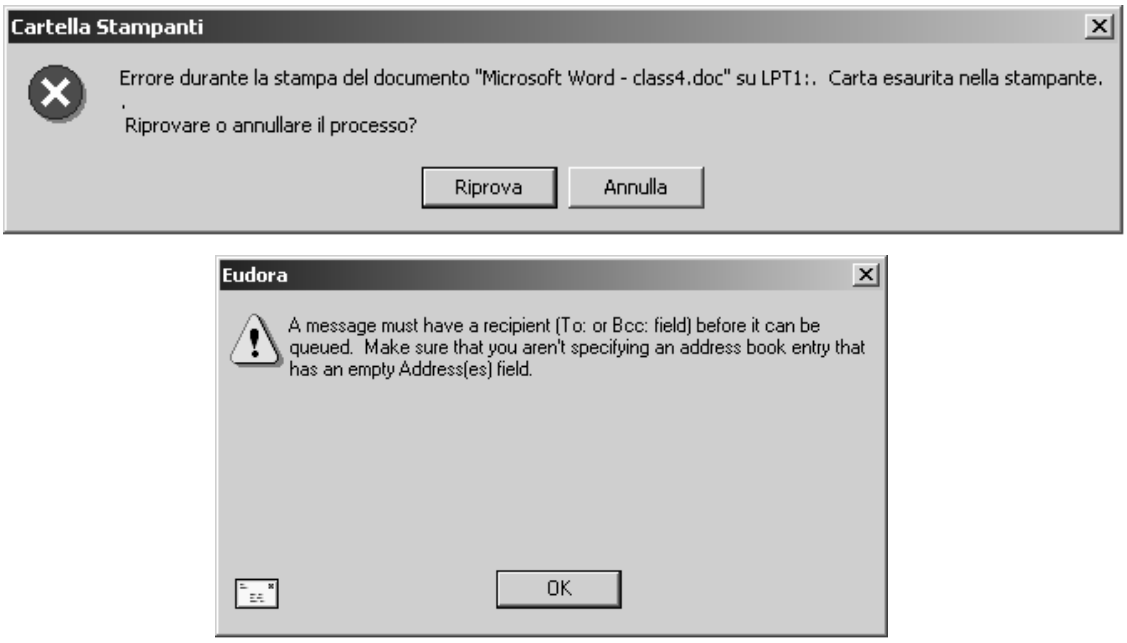

Figura 5.17: Messaggi di errore

Figura 5.18: Messaggio di 'attenzione'

b. messaggi 'amichevoli' sul risultato di una elaborazione

In altre fasi dell'interazione, Eudora 'partecipa' in modo empatico all'ipotizzato senso di delusione o di soddisfazione che l'utente proverà nel notare che non ha ricevuto o ha ricevuto posta.

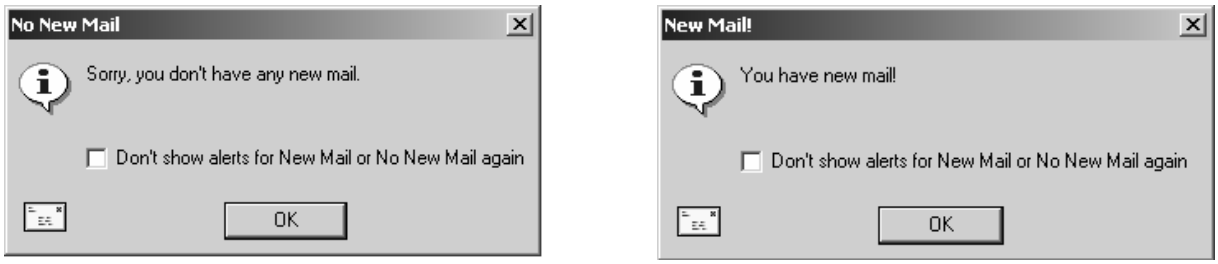

Figura 5.19: Messaggi sullo stato delle email

Esercizio 5.7

Come formuleresti il messaggio, nel caso in cui l'utente riceva un numero elevato di email di spamming?

b. messaggi 'fintamente amichevoli' su un'azione impropria compiuta dall'utente

Questo messaggio viene prodotto da Eudora quando si sta usando una versione free in cui è stata coperta la finestra di messaggi pubblicitari che l'applicazione sovrappone all'interfaccia in questo caso. Il messaggio è 'fintamente amichevole' perché assume che l'utente abbia fatto questo 'inadvertently' e gli chiede se davvero voleva farlo. Passa poi a giustificare perché i messaggi pubblicitari sono necessari e come Eudora cerchi di renderli il meno possibile 'intrusivi'. Conclude suggerendo di comprare la versione 'non free' per evitare questa intrusione. Il messaggio è molto lungo e la finestra è più grande di quella impiegata per il messaggio pubblicitario: l'effetto realmente voluto potrebbe essere quello di disturbare l'utente per 'convincerlo' a non coprire più, in futuro, il messaggio pubblicitario.

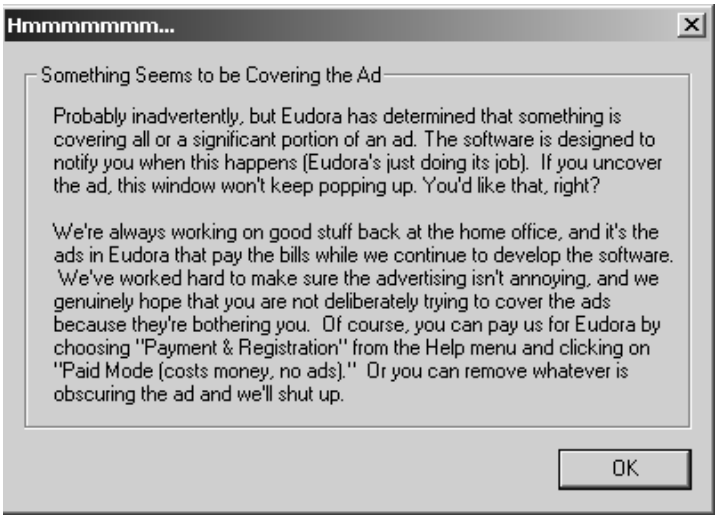

Figura 5.20: Messaggio di warning in Eudora

Figura 5.21: Messaggio dinamico sull'utilizzo della CPU in Windows