Modello a strati

Un sistema informativo è rappresentabile con un modello a strati:

Wan	Suolo pubblico, router, ADSL, HDSL, modem
Lan	Aree private, Ethernet, rame, fibra ottica, wifi, switch, cablaggio strutturato
Applicativi	App, Gestionale,
Software di base	Word-Processo, Foglio di Calcolo, pdf, browser, mail, runtime, compilatore, Visual studio, C#, Java
Sistema operativo	Windows, Linux, iOS, Android (GUI, Kernel, File system, Driver)
Firmware	Bios
Hardware	CPU RAM Bus HD ATX chipset USB VGA DVI HDMI

Nell'attività di programmazione si interagisce con tutti i livelli i del modello. In particolare vanno ricordati i seguenti dettagli.

Il firmware è software memorizzato in Rom. Il BIOS presente nei PC svolge due funzioni: controlla la funzionalità dell'hardware e se tutto è a posto avvia il bootstrap del sistema operativo.

Il kernel gestisce due risorse del PC: la memoria RAM e la CPU. La prima operazione che svolge è assegnare spazio in RAM al processo per caricare dati e istruzioni letti nella memoria di massa, di solito l'hard-disk. La seconda operazione è assegnare ad ogni processo caricato in RAM tempo CPU, sono utilizzati tempi dell'ordine del millisecondo per cui si ha l'impressione che decine di processi siano eseguiti contemporaneamente. La terza operazione avviene al termine del processo, la memoria RAM utilizzata dal processo è liberata e resa disponibile per successivi processi.
Il File-System è la parte del sistema operativo che gestisce le memorie di massa (HD SSD penne USB ...), il fyle system è organizzato in cartelle, sottocartelle e file.
La GUI è l'interfaccia grafica che permette all'operatore di interagire con mouse, tastiera e monitor. Esistono sistemi che non hanno un'interfaccia grafica e funzionano a linea di comando.
I driver sono l'interfaccia tra il sistema operativo e i dispositivi Hardware.

LAN e WAN usano protocolli in comune come l'IPV4. Una differenza importante è che le LAN hanno una gestione privata e non possono attraversare il suolo pubblico. Le WAN possono essere cablate sul suolo pubblico il cui utilizzo può essere concesso solo a operatori telefonici.

Installazione di un programma

Nell'installazione di un programma applicativo o di software di base si aggiungono al file-system della memoria di massa (HD) cartelle e sottocartelle che contengono i file dell'applicazione. Normalmente per l'installazione si usa un file di Setup (setup.exe).

Nell'installazione del sistema operativo prima di copiare i file che lo compongono si crea il file-system.

Lanciare un programma

Quando si lancia un programma il Kernel carica il programma leggendolo dal file system e copiandolo nella Ram. Ogni programma contiene byte, si distinguono istruzioni e dati. Quando si lancia il sistema operativo (avvio) è il bios che carica il sistema operativo dal file system alla ram.

Creare un progetto

Quando si crea un progetto con visual studio si crea nel file system una struttura di cartelle sottocartelle e file che conterranno le istruzioni e i dati dell'applicazione. Lavorando in modalità grafica (windows form) assume particolare importanza il disegno dell'interfaccia. Il programmatore inserisce oggetti nel programma interagendo con tastiera, mouse e monitor. Tali oggetti sono memorizzati in ram e sul monitor ne è visualizzata una parte.

Salvare un progetto

Con il comando salva il progetto è duplicato dalla ram nel file system di una unità di massa memorizzandolo in modo permanente. Dimenticare il comando salva (esempio in caso di blackout) significa perdere il lavoro fatto.

Lanciare un progetto

Quando si avvia un progetto da visual studio le istruzioni e i dati dell'applicazione sono passati dall'area dati dell'ambiente di sviluppo al kernel che li inserisce in ram assieme agli altri processi attivi.

Cosa succede quando si preme un tasto nella finestra codice di visual studio

Un segnale viene inviato dalla tastiera alla CPU.
La CPU riconosce la richiesta e tramite il driver della tastiera il sistema operativo legge il codice del tasto premuto e lo invia a visual-studio.
Visual studio utilizza il carattere premuto per aggiornare il testo di una istruzione nella ram.
Visual studio aggiorna la visualizzazione delle istruzioni nella finestra attiva.

Dichiarare una variabile

Dichiarando una variabile si decide che tipo di informazioni (numeri, stringhe, ...) si potranno associare a un nome di rifermento alfanumerico. In memoria non esiste nessun byte, il nome è gestito da visual studio che lo può utilizzare per il controllo della sintassi delle istruzioni. Esempio può segnalare l'assegnazione errata di una stringa ad una variabile numerica.

Istanziare una variabile

Istanziando una variabile si riserva uno spazio dati in ram che le istruzioni potranno utilizzare per memorizzare dei valori specifici. Con il termine null si indica il fatto che i byte di memoria non hanno mai ricevuto un valore.

Assegnare una variabile

Assegnando un valore a una variabile con un nome specifico una istruzione inserisce un valore nello spazio dati in ram.

Programmi e istruzioni

Scrivendo un'istruzione si aggiunge un comando nel programma per svolgere un compito specifico. Le istruzioni sono eseguite nella maggior parte dei casi in ordine sequenziale. Negli esercizi normalmente si distingueranno:

1. Dichiarazioni di variabili
2. Lettura input con conversione dei dati inseriti tramite l'interfaccia grafica a variabili.
3. Algoritmo utilizzando le variabili che rappresentano l'input si calcolano le variabili risultato che sanno visualizzate successivamente.
4. Scrittura output con conversione dei risultati in un formato utilizzabile nell'interfaccia grafica.

OOP (object oriented programming)

Nella programmazione orientata agli oggetti si definiscono classi di oggetti caratterizzate da: proprietà, metodi ed eventi.

Istanziare un oggetto

Quando si istanzia un oggetto di una classe avvengono le seguenti operazioni:

- si riserva uno spazio nell'area dati in cui possano essere memorizzate le proprietà dell'oggetto (nome, testo, top, left, dimensioni, font, colore, …)
- si riserva una spazio nell'area dati in cui possano essere memorizzate le variabili locali dei metodi
- si caricano le istruzioni dei metodi
- si riserva una spazio nell'area dati in cui possano essere memorizzate le variabili locali degli eventi
- si caricano le istruzioni degli eventi.

INT

Le variabili di tipo int sono memorizzate in gruppi di byte.

numero di byte	signed (minimo … massimo)	unsigned (minimo … massimo)
1 (byte)	-128 … 127	0 … 255
2 (int)	-32K … 32K	0 … 64K
4 (long)	-2G … 2G	0 … 4G
8	$-(2G)^2$ … $(2G)^2$	0 … $(4G)^2$

FLOAT

Per memorizzare un numero con cifre decimali si assegnano un certo numero di bit per l'esponente e un certo numero di bit per la mantissa e un bit per il segno.

numero di byte	bit esponente	bit mantissa	minimo … massimo
4 (float)	8	23	$3.4*10^{-38}$ … $3.4*10^{38}$
8 (double)	11	52	$1.7*10^{-308}$ … $1.7*10^{308}$

STRING

Le stringhe sono formate da gruppi di caratteri. Le codifiche più comuni sono ASCII e Unicode. Ascii prevede una codifica di un byte per ogni simbolo per cui si possono gestire massimo 255 simboli diversi. Unicode utilizza due byte per ogni simbolo per cui si possono gestire 64K simboli diversi.

Le stringhe sono adatte per la gestione dell'interfaccia grafiche in quanto formate da caratteri sono facilmente leggibili dalle persone che utilizzano il PC. Al contrario le variabili numeriche sono memorizzate in formato binario che è più efficiente per l'esecuzione di calcoli ma risulta interpretabile con estrema difficolta dalle persone.

VETTORI

I vettori sono utilizzati per memorizzare con un unico nome più elementi individuati da un indice numerico. Da utilizzare con le strutture iterative (cilcli). Esempio i voti di una pagella possono essere memorizzato con una serie di variabili numeriche (votoItaliano, votoStoria, votoInglese, votoMatematica, …, votoCondotta) o con un vettore (voto(0) per italiano, voto(1) per storia, voto(2) per inglese, voto(3) per matematica, …, voto(9) per la condotta).

IF

Si compone di quattro parti

- If (condizione)
  - Blocco istruzioni eseguite per condizione vera
- Else
  - Blocco istruzioni eseguite per condizione falsa.

L'else e il secondo blocco di istruzioni sono opzionali.

FOR

Il ciclo for si compone di due parti

- For (inizializzazione, condizione, incremento)
  - Blocco istruzioni ripetute per condizione vera

L'inizializzazione è eseguita la prima volta che il programma arriva all'istruzione for.
La condizione è verificata ogni volta prima di eseguire il ciclo.
L'incremento è fatto alla fine di ogni ciclo prima di verificare la condizione

Esercizio cicli nidificati

```
For (i=0; i<3; i++)
        For (j=0; j<2; j++)
                If (i==0)
                        msgBox("denominatore nullo")
                else
                        msgBox(convert.toString(j/i))
```

La tabella seguente descrive la successione delle istruzioni

Istruzione	i	j	I<3	J<2	I==0	msgBox
For (i=0; i<3; i++)	0		vero			
For (j=0; j<2; j++)		0		vero		
If (i==0)					vero	
msgBox("denominatore nullo")						Denominatore nullo
For (j=0; j<2; j++)		1		vero		
If (i==0)					vero	
msgBox("denominatore nullo")						Denominatore nullo
For (j=0; j<2; j++)		2		falso		
For (i=0; i<3; i++)	1		vero			
For (j=0; j<2; j++)		0		vero		
If (i==0)					falso	
msgBox(convert.toString(j/i))						0
For (j=0; j<2; j++)		1		vero		
If (i==0)					falso	
msgBox(convert.toString(j/i))						1
For (j=0; j<2; j++)		2		falso		
For (i=0; i<3; i++)	2		vero			
For (j=0; j<2; j++)		0		vero		
If (i==0)					falso	
msgBox(convert.toString(j/i))						0
For (j=0; j<2; j++)		1		vero		
If (i==0)					falso	
msgBox(convert.toString(j/i))						0,5
For (j=0; j<2; j++)		2		falso		
For (i=0; i<3; i++)	3		falso			

funzioni e parametri

Le funzioni sono gruppi di istruzioni che svolgono compiti specifici. Alle funzioni è possibile passare parametri che contengono valori da utilizzare durante il funzionamento. Il funzionamento produce un risultato che è restituito con l'istruzione return.

```
public double somma(double a, double b)
        double r;
        r = a +b
        return r
```

Nell'esempio si passano due parametri per valore. Le variabili a e b saranno dei duplicati che conterranno i valori utilizzati nella funzione di chiamata somma(N1, N2). Il contenuto della variabile N1 è duplicato nella variabile a, il contenuto della variabile N2 è duplicato nella variabile b.

Supponiamo di avere due textBox di input (tbxN1 e tbxN2) che contengono due stringhe ("12" e "22") che rappresentano due numeri da sommare, un bottone (btnSomma) usato per attivare il calcolo della somma, una textBox di output (tbxN3) per visualizzare i risultato.

All'interno dell'evento click del bottone si chiama la funzione somma.

```
btnSomma.click(...)
        double N1, N2, N3                   //dichiarazione di tre variabili numeriche

        N1 = convert.toDouble(tbxN1.text)   //lettura degli input
        N2 = convert.toDouble(tbxN2.text)

        N3 = somma(N1, N2)                  //chiamata funzione somma

        tbxN3.text = convert.toString(N3)   // scrittura degli output
```

Istruzione	N1	N2	N3	a	b	r	tbxN1.text	tbxN2.text	tbxN3.text
double N1, N2, N3	null	null	null	ne	ne	ne	"12"	"22"	
N1 = convert.toDouble(tbxN1.text)	12								
N2 = convert.toDouble(tbxN2.text)		22							
.... somma(N1, N2)									
public double somma(double a, double b)				12	22				
double r						null			
r = a +b						34			
return r									
N3 = ...			34	ne	ne	ne			
tbxN3.text = convert.toString(N3)									"34"

null = valore nullo.
ne = non esiste in memoria.
"xx" = stringa