# I - Concetti Introduttivi

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

Il sistema operativo è un componente software di un sistema di elaborazione il cui compito principale è quello di controllare l'esecuzione dei programmi applicativi e di agire come intermediario tra questi e l'hardware, con lo scopo di facilitare l'uso e, al tempo stesso, di garantire che tale uso sia effettuato in maniera efficace ed efficiente.

Fornendo dunque una interfaccia alle applicazioni, mostra loro una macchina virtuale, più semplice da gestire, permettendo una maggiore portabilità dei programmi e garantendo anche la protezione e la sicurezza dei dati dei vari utenti.

In particolare un sistema deve garantire la riservatezza (privacy) e la protezione delle informazioni.

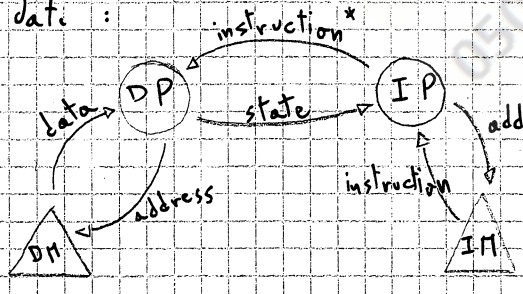
L'interfaccia fornita dal sistema è detta API (Application Programming Interface).

———— o ————

## Classificazione dei calcolatori

La Tassonomia di Flynn classifica le architetture dei calcolatori a seconda della molteplicità dei flussi di istruzioni e dei flussi dei dati che possono gestire; ogni computer può essere quindi classificato come:

- Single Instruction Single Data: una singola unità esegue un singolo flusso di dati:



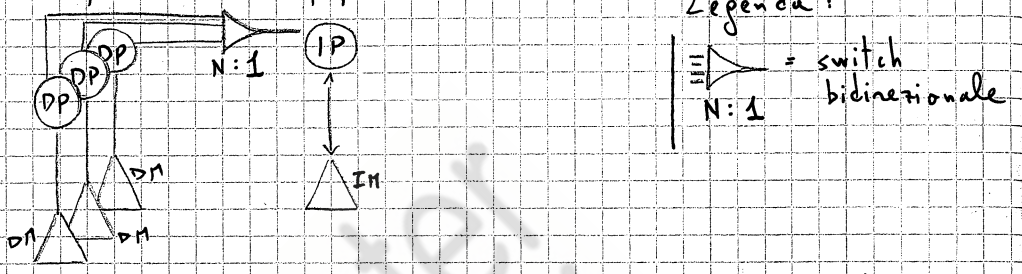

Legenda:
- ○ = insieme
- △ = gerarchia
- ↖ = bus

Nella figura, un solo Processore delle Istruzioni è collegato con un solo Processore dei Dati (con le rispettive memorie), quindi viene eseguita una sola istruzione per volta che opera su un solo dato.

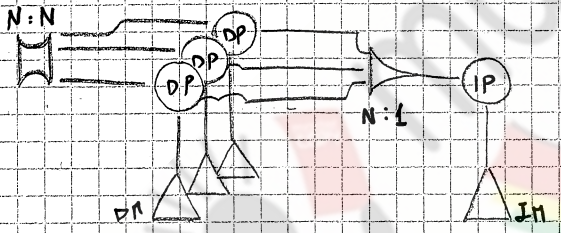
Sostanzialmente, l'IP è la parte dedicata alla Fetch, Decode, Execute, ecc e tutti i registri che supportano il ciclo del processore, il DP comprende la ALU e tutti i registri generali.

L'architettura mostrata in figura riporta due gerarchie di memorie distinte (istruzioni e dati); è detta architettura di Harvard e ha il vantaggio di eliminare la competizione per l'uso del bus.
Comunque, anche nei computer che non la implementano alla lettera si è capaci di distinguere i due flussi logici, quindi una certa distinzione logica resta.

* Single Instruction Multiple Data (pipeline)




Molte ALU accedono a altrettante memorie dati; in questo modo però le DP non possono scambiare informazioni. Uno schema migliore è




Questo schema non ha limiti veri e propri di numero di DP, quanto di banda passante e di "routing": non si può impiegare un bus per interconnettere i DP, poiché esso implica un accesso in mutua esclusione.

Questa struttura è chiamata Array Processor (processore vettoriale) o Array Sistolici (orientati cioè al flusso dei dati; gli stessi dati vengono impiegati in più operazioni di fila, modificati usando uno schema a pipeline).

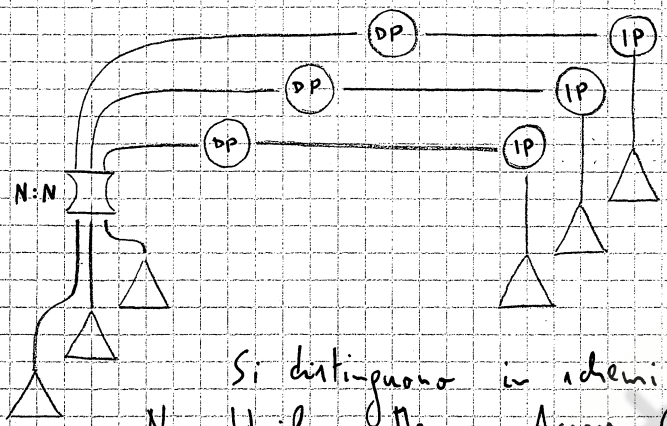
Permettono inoltre l'applicazione simultanea di stesse istruzioni su vettori di dati.

Il sorgente per questi processori non deve essere rimodellato, basta una ricompilazione del codice prodotto per processori SISD.

◊ Multiple Instruction Single Data

non esistono computer con questa architettura.

◊ Multiple Instruction Multiple Data - Shared Memory

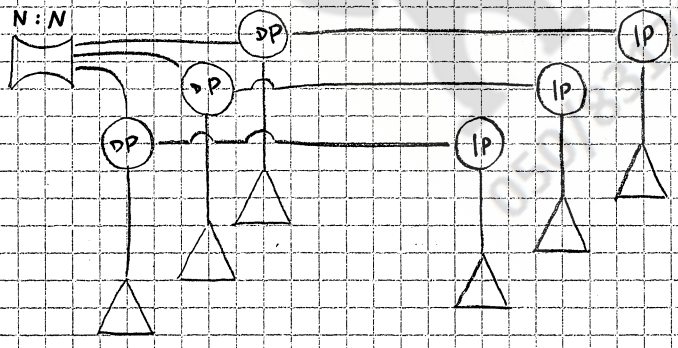



Un tipico computer desktop moderno, purtroppo questo schema scala poco (senza dover essere modificato), quindi il numero dei processori resta limitato.

Si distinguono in schemi con Uniform Memory Access e con Non Uniform Memory Access (di questi ultimi vi è un'ulteriore distinzione tra sistemi con o senza cache).

È lo schema Multi Processor; come tale ha problemi per via della condivisione della memoria e delle periferiche, perciò è indirizzata una certa multiprogrammazione su ogni processore anche in questo schema. Il problema diventa mantenere il carico sui singoli processori bilanciato e occupare il più possibile tutti i core.

◊ MIMD - Distributed Memory




Schema a Multi Computer, detto anche Messively Parallel Processing, tipico delle moderne applicazioni distribuite (via Internet); se si stacca una delle unità di elaborazione, questa rimane comunque un computer completo.

Per l'elevata complessità, la Rete non può però essere del tutto schematizzata come un computer.

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745

Si indica con speed up il rapporto $T_1 / T_N$, dove $T_1$ è il tempo di esecuzione di una applicazione su una macchina SISD e $T_N$ il tempo su una macchina SIMD o MIMD. Si cerca di ottenere uno speed up in genere altamente elevato ($T_N \ll T_1$).

> Il porting di una applicazione in un formato puramente MIMD è un lavoro molto oneroso; conviene spesso convertire una app. tradizionale in una eseguibile con molti task (il più possibile paralleli e bilanciati).

In generale, esiste una formula, detta Legge di Amdahl, che permette di ricavare lo speed up:

$$S_u = \frac{T_1}{T_{seq} + \frac{T_1 - T_{seq}}{N}}$$ si cerca in genere di avere $S_u < \frac{T_1}{T_{seq}}$

dove $T_{seq}$ rappresenta la parte del processo che deve essere necessariamente eseguita sequenzialmente (e dunque non può beneficiare della parallelizzazione).

> Altre informazioni e precisazioni disponibili su
> → it.wikipedia.org/wiki/Tassonomia_di_Flynn (e voci correlate) e su
> → it.wikipedia.org/wiki/Legge_di_Amdahl .

---

## Modelli Strutturali di Sistemi Operativi

I primi sistemi erano costituiti da un unico programma, privo di una vera e propria suddivisione dello stesso in moduli distinti. Tali sistemi sono detti Monolitici e sono sostanzialmente un insieme di procedure di servizio a ciascuna delle quali corrisponde una chiamata di sistema.

In seguito, per permettere una maggiore modularità, si passò ai sistemi Modulari, nei quali ogni modulo ha una sua interfaccia e una sua implementazione separata. Da qui, si passò a distinguere i moduli in due categorie, le procedure di servizio e le procedure di utilità.

Queste ultime non sono visibili ai programmi e sono di supporto a quelle di servizio.

Aumentando in complessità, si è ritrovati a dover sviluppare sistemi operativi per livelli gerarchici di astrazione, nei quali il livello più basso è detto Kernel (nucleo); i livelli successivi si appoggiano non sull'hw della macchina, ma sulla macchina virtuale definita dall'interfaccia del livello precedente.

Per aumentare la dinamicità e la estendibilità del sistema, venne introdotta la struttura a Microkernel, nella quale solo un minimo dei meccanismi di gestione delle risorse viene svolto nel Kernel, mentre la maggior parte delle operazioni viene delegata a una serie di processi applicativi di sistema (daemon).

‖ L'insieme dei meccanismi costituisce il microkernel, che gira a stato privilegiato; tutte le strategie vengono implementate invece dai programmi di sistema (procmi).

La richiesta di un servizio al sistema comporta quindi uno scambio di messaggi tra processi. Per via di problemi legati alle prestazioni, in genere quasi tutti i sistemi a microkernel sono più forme ibride, con alcune funzionalità sviluppate nel kernel (Windows NT, Mac OS X).

Poco ~~diffuso~~ diffuso è un altro modello strutturale, quello Client-Server, interamente basato su strutture distribuite con ognuna un proprio Kernel che si scambiano informazioni sotto forma di messaggi.

———— o ————

MASTER COPY
Tel. 050 8312126
Cell. 388 9837745