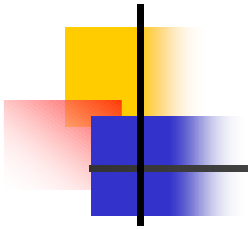
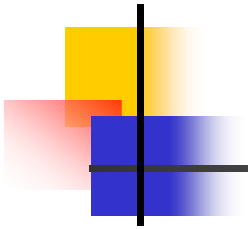


Gestione del processore e dei processi

- Il processore è la componente più importante di un sistema di elaborazione e pertanto la sua corretta ed efficiente gestione è uno dei compiti principali di un sistema operativo
- Il ruolo del processore è quello di *eseguire programmi*
- Consideriamo la seguente definizione di **processo**:

un processo è un programma in esecuzione

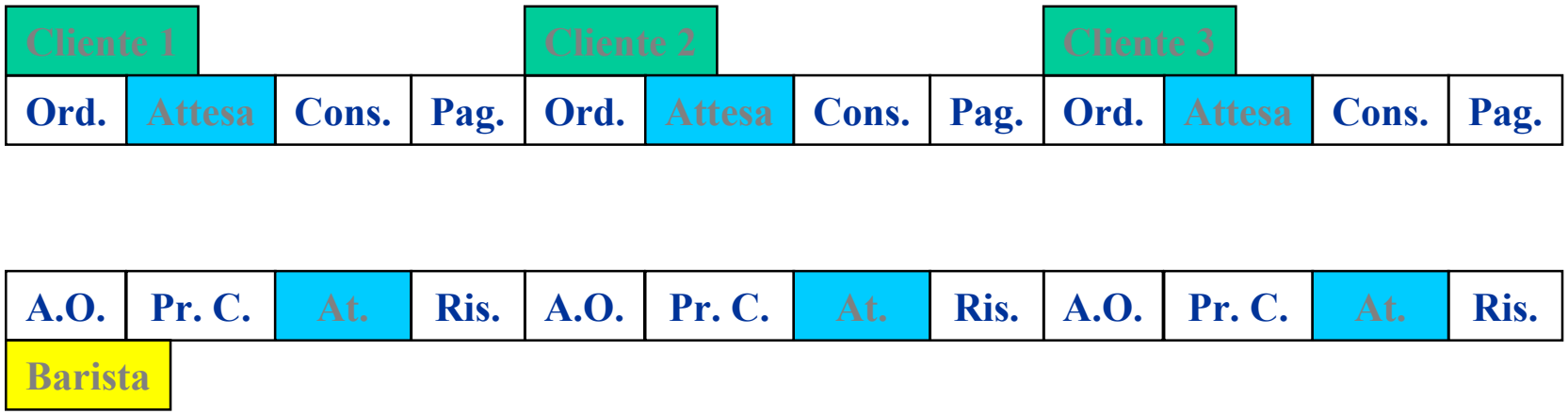


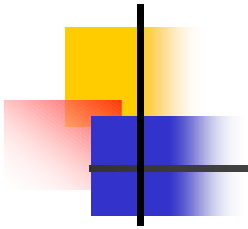
Gestione del processore e dei processi

- Uno dei compiti principali del sistema operativo riguarda la gestione dei processi
- Per il momento assumiamo di trattare sistemi mono-programmati (anche detti **mono-tasking**)
- In sistemi di questo tipo è possibile eseguire un solo programma alla volta: i programmi devono essere eseguiti in modo sequenziale e si può mandare in esecuzione un programma solo quando quello precedente ha terminato l'esecuzione (es. MS/DOS)

Gestione del processore e dei processi

- Supponiamo che il nostro sistema sia un bar in cui un barista serve diversi clienti
- Il barista è il corrispondente del processore, i clienti sono l'equivalente dei processi da eseguire
- Esecuzione mono-tasking:





Gestione del processore e dei processi

- Quello che viene evidenziato dall'esempio precedente è vero in generale:

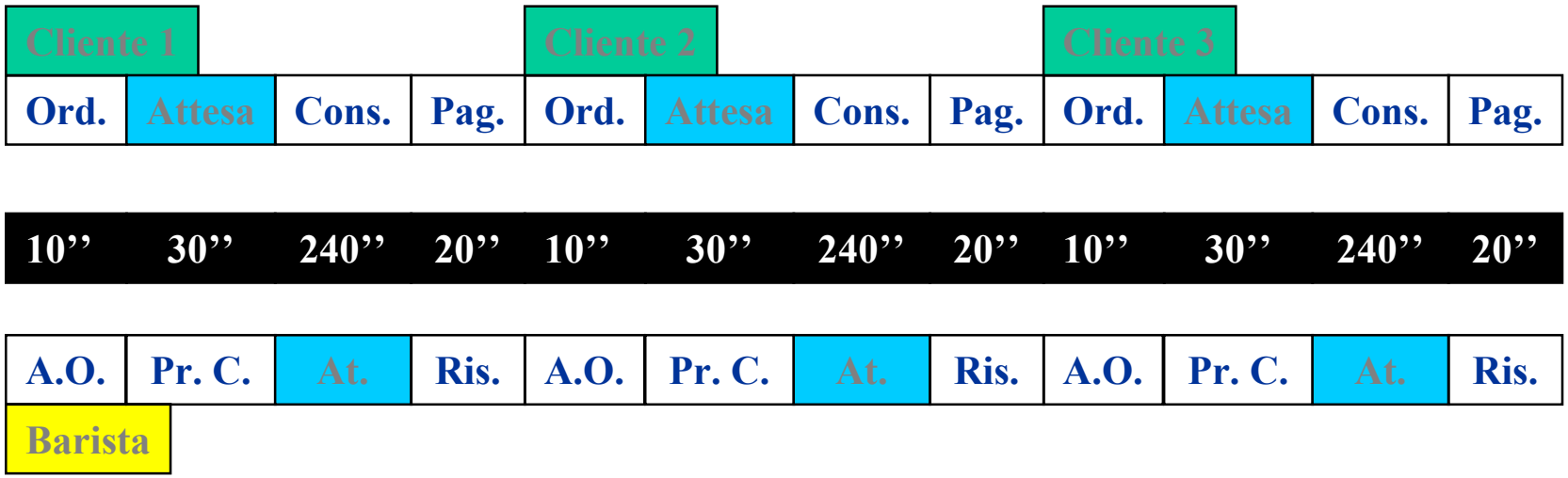
  qualunque processo P alterna fasi di esecuzione a fasi in cui è bloccato in attesa di qualche evento esterno

- Un processo può essere in attesa che sia terminata un'operazione di input di dati oppure in attesa di poter usare una risorsa in quel momento occupata

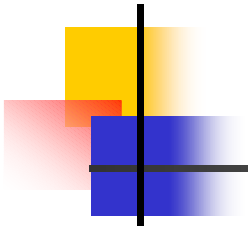
Gestione del processore e dei processi

- Se si confrontano la velocità di elaborazione di un calcolatore (quindi il tempo in cui un processo è in esecuzione) e i tempi di lavoro delle periferiche di input/output, o addirittura i tempi di reazione umani, risulta che i secondi sono di molti ordini di grandezza maggiori dei primi
- Un elaboratore medio esegue un'istruzione di un programma in un tempo che è dell'ordine di grandezza di un milionesimo di secondo, mentre le periferiche di input/output operano in modo molto più lento, e i tempi di reazione umani per fornire dei dati in input sono addirittura dell'ordine dei secondi
- In un'esecuzione sequenziale, mentre il processo attivo è bloccato in attesa di eventi esterni, il processore rimane inattivo, in uno stato chiamato **idle**, e risulta pertanto sotto-utilizzato

Gestione del processore e dei processi



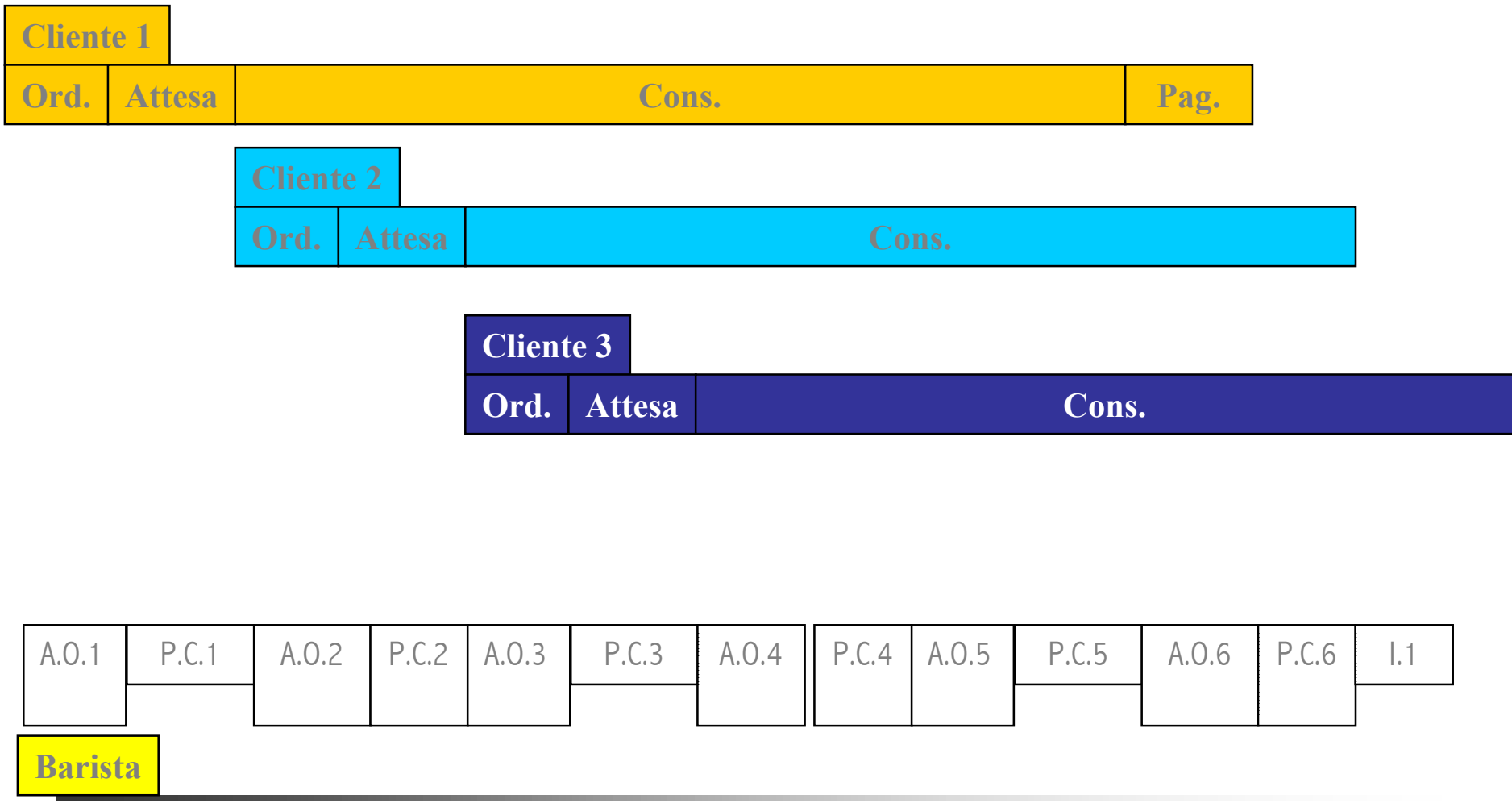
- Il tempo impiegato per servire un cliente è pari a 300"
- Durante questo tempo il barista attende (senza fare altro) per un tempo pari a 240"
- Il tempo di attesa (non produttivo) è **80%** del tempo totale

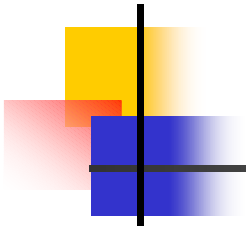


Gestione del processore e dei processi

- Una soluzione ragionevole per migliorare sia la "produttività" del barista che la qualità del servizio (i clienti arrivano e si mettono in coda aspettando di essere serviti) potrebbe essere quella di servire i più clienti *contemporaneamente*
- Ogni volta che che un cliente inizia la consumazione del suo caffè, il barista, invece di restare inattivo può iniziare il servizio del prossimo cliente

Gestione del processore e dei processi





Gestione del processore e dei processi

- Si noti che il termine

  contemporaneamente

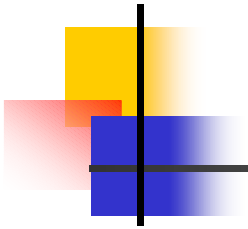
  è sempre scritto in corsivo
- Infatti, poiché vi è un solo barista (un solo processore nel caso di un computer) il servizio non è realmente eseguito in parallelo ma avviene alternando il servizio tra i vari clienti
- In questo modo i tempi di inattività del barista sono eliminati

Gestione del processore e dei processi

- L'idea illustrata mediante l'esempio precedente è l'idea di base del **multi-tasking**:

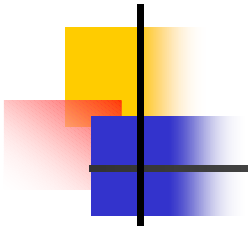
  più programmi vengono eseguiti *contemporaneamente* sullo stesso processore. Il numero di processi attivi viene detto **grado di multi-programmazione** del sistema

- Dal punto di vista del processore, in ogni istante vi è un solo processo in esecuzione
- Se l'alternanza tra i processi è frequente (ad esempio ogni 10 millisecondi), l'utente ha l'impressione che l'esecuzione dei programmi sia veramente simultanea



Gestione del processore e dei processi

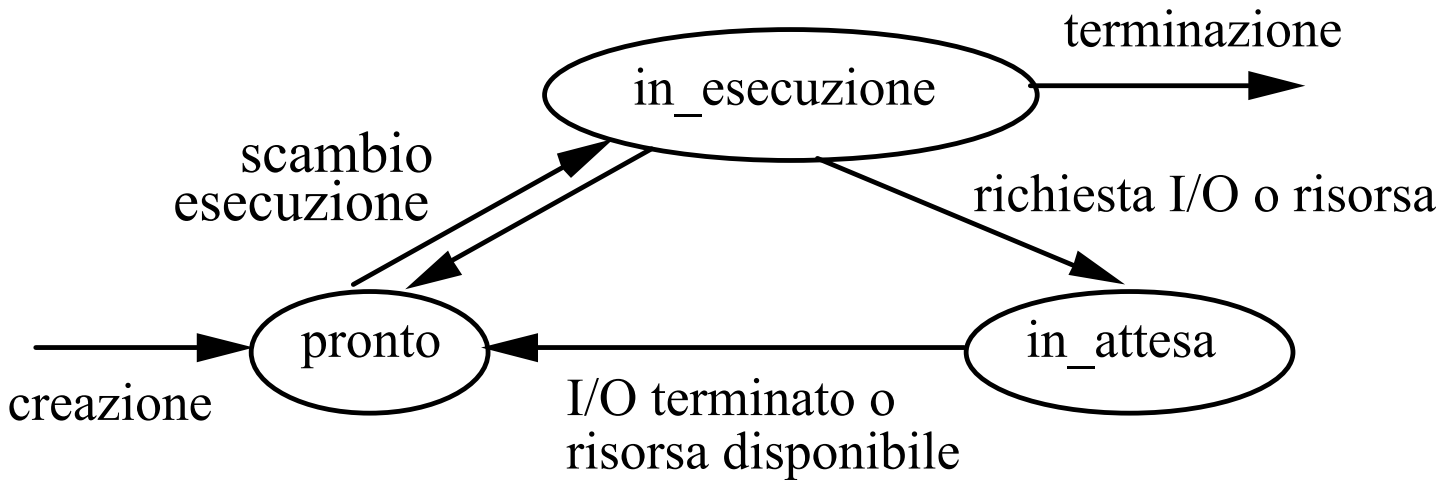
- A livello macroscopico si ha quindi l'impressione della contemporaneità, mentre a livello microscopico si ha una semplice alternanza sequenziale molto veloce
- Si noti che dal punto di vista di ogni singolo programma il tempo di esecuzione, cioè il tempo che intercorre tra l'inizio e la fine del processo, risulta aumentato rispetto al caso mono-programmato a causa dell'alternanza con gli altri processi

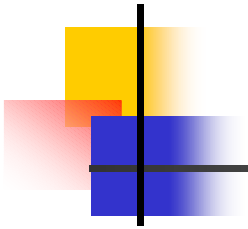


Gestione del processore e dei processi

- Un processo può trovarsi in tre diversi **stati**:
  - in **esecuzione**, quando sta utilizzando il processore;
  - in **attesa** (**bloccato**), quando è in attesa del verificarsi di un evento esterno (terminazione di un'operazione di input/output o altro)
  - **pronto**, quando è potenzialmente in condizione di poter utilizzare il processore che è occupato da un altro processo

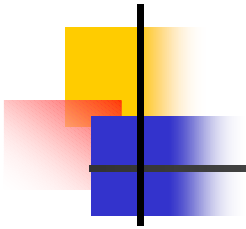
Gestione del processore e dei processi





Gestione del processore e dei processi

- Quando processo viene creato (alla richiesta da parte di un utente di eseguire un programma) viene messo nello stato di *pronto* e in tale stato rimane fino a quando non arriverà il suo turno
- Non appena un processo pronto viene selezionato, passa nello stato di *esecuzione*
- Un processo può abbandonare lo stato di *esecuzione* per tre diverse ragioni

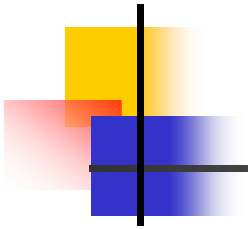


Gestione del processore e dei processi

- **Terminazione**: il processo termina la sua esecuzione e abbandona il sistema
- Richiesta di un'**operazione di input/output** o di una **risorsa** occupata. In questo caso il processo passa allo stato di *attesa*, il processore viene liberato e può essere concesso ad un altro processo *pronto*
- **Cambio di esecuzione**: per realizzare in modo equo l'alternanza tra i vari processi, in certi casi può essere opportuno fermare un processo, rimetterlo nello stato di *pronto* e concedere il processore ad un altro processo

Gestione del processore e dei processi

- Il multi-tasking funziona efficacemente se il S.O. è in grado di offrire alcune assicurazioni
- Il cambio di contesto deve essere invisibile al processo. Nell'esempio del bar i clienti non si accorgono che il barista serve un gruppo di clienti contemporaneamente
- Quando il barista riprende a servire un cliente **il servizio viene ripreso esattamente nel punto in cui era stato interrotto**
- Il barista ha un cronometro e ogni ***n*** secondi sospende il servizio che sta effettuando e passa a servire un altro cliente



Gestione del processore e dei processi

- La politica di gestione in cui il barista divide il suo tempo tra i vari clienti assegnando a ciascuno un certo intervallo di tempo è implementata nei S.O. multi-tasking e prende il nome di **Round-Robin**
- La politica di gestione in cui il barista serve in modo sequenziale i vari clienti viene chiamata **FIFO**