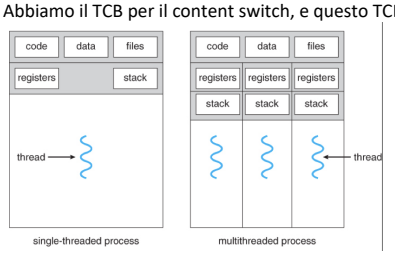
Multithreading teorico

Monday, 20 March 2023 13:26

- Chiamati processi leggeri
  Essi sono distinte operazione che sono dentro ad uno stesso processo
  Quindi:
  - Si può condividere la memoria di default siccome la memoria globale è condivisa
  - E' impossibile usare la strategia via messaggi siccome siamo dentro lo stesso processo

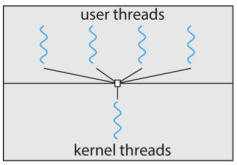
  Abbiamo il TCB per il content switch, e questo TCB è collegato ad un TCB



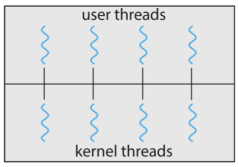
- Vantaggi:
  - Condivisione dei dati: memoria condivisa
  - Economia: creare un thread è più leggere
  - Migliori tempi risposta: l'applicazione può eseguire operazione onerose in un thread, e l'interfaccia utente in un altro
  - Scalabilità: il multithreading aumenta il parallelismo di un'applicazione
  - Ottimizzazione risorse: I/O condivisi, quindi no context switch se 2 thread hanno bisogno della stessa I/O
- Svantaggi:
  - Bilanciamento compiti
  - Suddivisione dati per massimizzare accesso parallelo
  - Dipendenza tra i dati dei task per stabilire rispettive sincronizzazioni
  - Testing/Debugging

- I thread si suddividono in:
  - Utente, disponibile dagli utenti offerti dalla libreria thread
  - Kernel, implementati nativamente dal kernel, ed una porziona da quella utente

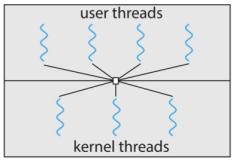
  Siccome un sottoinsieme dei thread kernel sono thread utenti, ci troviamo in una condizione dove dobbiamo decidere a chi dare il thread, ed a chi non darlo visto che, è possibile che abbiamo più richieste di utilizzo thread di quanti thread utente abbiamo.
  Detto questo, sono adottate delle strategie per gestire processo -> threads:
  - 1 solo thread kernel ha tutti i thread utenti



  - Uno-a-uno
    Ogni thread kernel ha 1 thread utente

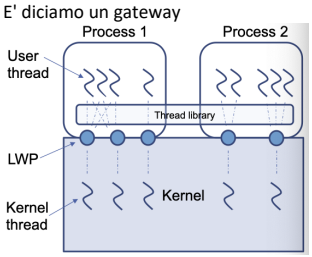


  - Molti-a-molti
    Non te lo saprei manco spiegare



    Sembra un ragno!

  Le librerie utenti, quando vogliono un thread chiedono al LWP, lightweight process
  E' diciamo un gateway



  L'attivazione dello scheduler l'ho saltata, ma non credo serve per questa materia
  Buona fortuna per sistemi e reti! Siccome le slide le hanno fottute da loro