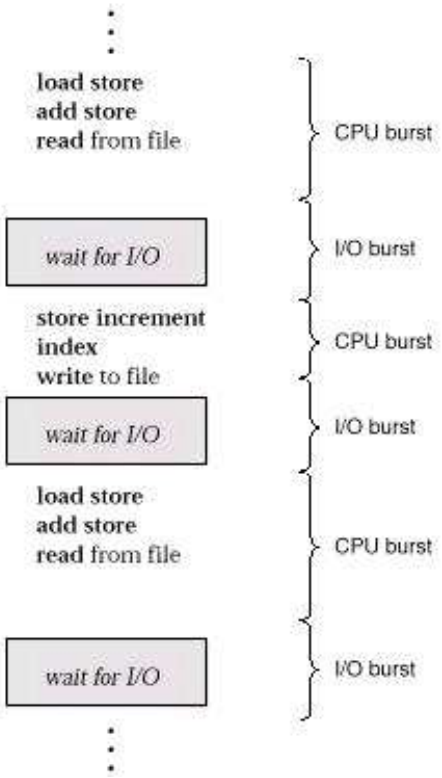
Planowanie przydziału procesora

Pojęcia podstawowe

Wieloprogramowanie jako narzędzie maksymalnego wykorzystania procesora.

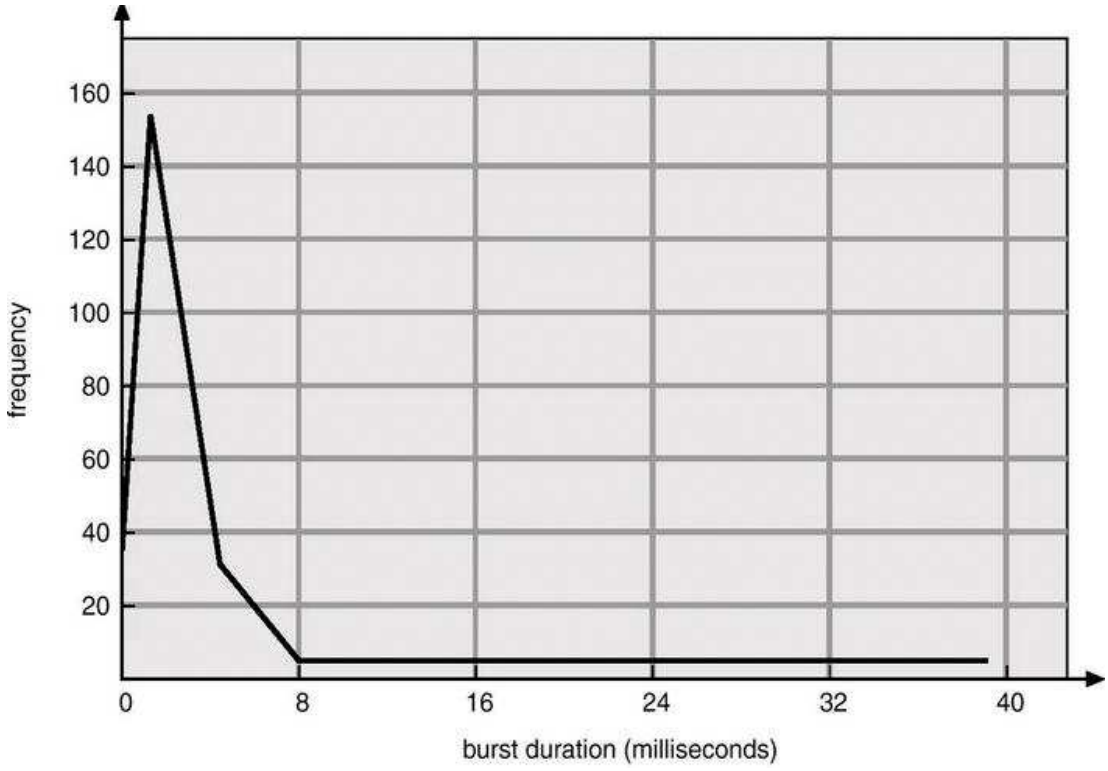
Wykonanie procesu — naprzemienne występowanie faz procesora i wejścia/wyjścia.



Rozkład czasów faz procesora zależy istotnie od procesu i sprzętu ale krzywa częstości ma zazwyczaj kształt zbliżony do pokazanej.

Planista przydziału procesora

Spośród procesów znajdujących się w pamięci wybiera jeden, któremu ma zostać przydzielony procesor.



Planista podejmuje decyzje w następujących przypadkach:

1. proces przeszedł ze stanu wykonywania do stanu oczekiwania,
2. proces przeszedł ze stanu wykonywania do stanu gotowości (np. przerwanie),
3. proces przeszedł ze stanu czekania do stanu gotowości (np. zakończenie we/wy),
4. proces zakończył działanie.

W przypadkach 1 i 4 planowanie odbywa się bez wywłaszczania — procesor został oddany dobrowolnie.

W przypadkach 2 i 3 możliwe jest planowanie z wywłaszczeniem.

Dispatcher (ekspedytor)

Faktycznie przekazuje kontrolę nad procesorem procesowi wskazanemu przez planistę krótkoterminowego, to znaczy:

- przełącza kontekst,
- przełącza tryb pracy na tryb użytkownika,
- wykonuje skok do odpowiedniego miejsca wznawianego programu.

Opóźnienie ekspedycji (dispatch latency).

Kryteria planowania

- wykorzystanie CPU — jaką część czasu procesor jest zajęty,
- przepustowość — liczba procesów zakończonych w jednostce czasu,
- czas cyklu przetwarzania — od chwili pojawienia sie zadania do jego zakończenia,
- czas oczekiwania w kolejce procesów gotowych,
- czas reakcji (odpowiedzi) — w pracy interakcyjnej.

Kryteria:

- maksymalizacja wykorzystania CPU,
- maksymalizacja przepustowości,
- minimalizacja czasu cyklu przetwarzania,
- minimalizacja czasu oczekiwania,
- minimalizacja czasu reakcji.

Algorytmy przydziału procesora

Ilustrując działanie algorytmów dla uproszczenia będziemy rozważać tylko jedną (najbliższą) fazę pracy procesora.

Miarą stosowaną w porównaniach będzie średni czas oczekiwania.

FCFS: pierwszy zgłoszony — pierwszy obsłużony

Ten, kto pierwszy złożył zamówienie, ten pierwszy dostaje.

Nie stosuje się wywłaszczania.

proces	faza procesora
P1	24
P2	3
P3	3

1. Niech procesy pojawią się w kolejności: P1, P2, P3.

Czas oczekiwania: P1 - 0, P2 - 24, P3 - 27.

Średni czas oczekiwania: 17.

2. Procesy pojawiają się w kolejności: P2, P3, P1.

Czas oczekiwania: P1 - 6, P2 - 0, P3 - 3.

Średni czas oczekiwania: 3.

Efekt konwoju — mniejsze wykorzystanie procesora i urządzeń.

Nie nadaje się do systemów z podziałem czasu.

SJF: najpierw najkrótszy

Procesor dostaje ten, którego najbliższa faza procesora jest najkrótsza (dla równych faz - FCFS).

Dla procesów:

proces	faza procesora
P1	6
P2	8
P3	7
P4	3

algorytm daje średni czas oczekiwania - 7.

Zastosowanie algorytmu FCFS - 10,25.

Algorytm *optymalny* ze względu na średni czas oczekiwania dla danego zbioru procesów.

Dwie wersje:

- bez wywłaszczania — przydzielony procesor nie może zostać odebrany,
- z wywłaszczaniem — w momencie pojawienia się nowego procesu następuje wywołanie planisty (wersja zwana SRTF — najkrótszy czas pozostały najpierw)

Rozważmy zbiór procesów:

proces	czas przybycia	faza procesora
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

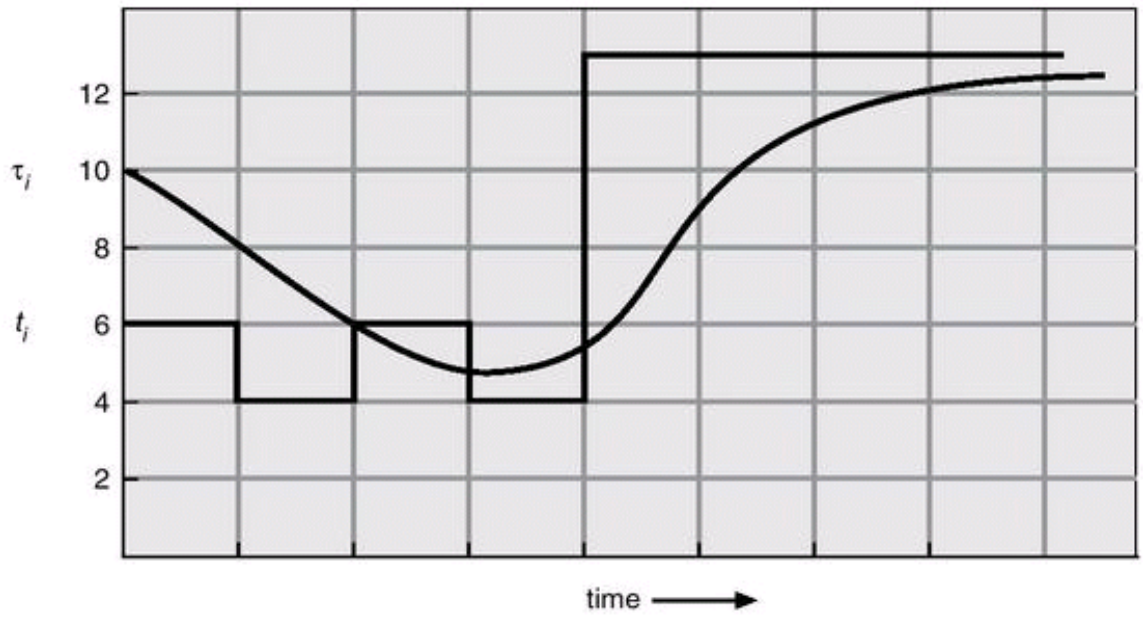
- wersja bez wywłaszczania daje średni czas oczekiwania - 4.
- wersja z wywłaszczaniem - 3. (słuchacz do tablicy !!!)

W praktyce algorytmu nie da się stosować w wersji oryginalnej — nie ma jak określać długości kolejnych zamówień.

Przybliżamy następną fazę na podstawie długości faz poprzednich.

- t_n — faktyczna długość n-tego cyklu procesora,
- τ_{n+1} — przewidywana długość kolejnego cyklu,
- dla α takiego, że $0 \leq \alpha \leq 1$ definiujemy:

$$\tau_{n+1} = \alpha t_n + (1 - \alpha)\tau_n$$



CPU burst (t_i)		6	4	6	4	13	13	13	...
"guess" (τ_i)	10	8	6	6	5	9	11	12	...

- $\alpha = 0$
  - $\tau_{n+1} = \tau_n$

  - niedawna historia nie ma znaczenia (chwilowe zaburzenia).
- $\alpha = 1$
  - $\tau_{n+1} = t_n$
  - liczy się tylko najnowsza faktyczna faza.
- $0 < \alpha < 1$
  - $\tau_{n+1} = \alpha t_n + (1 - \alpha)\alpha t_{n-1} + ...$
  - każdy kolejny składnik ma mniejszą wagę.

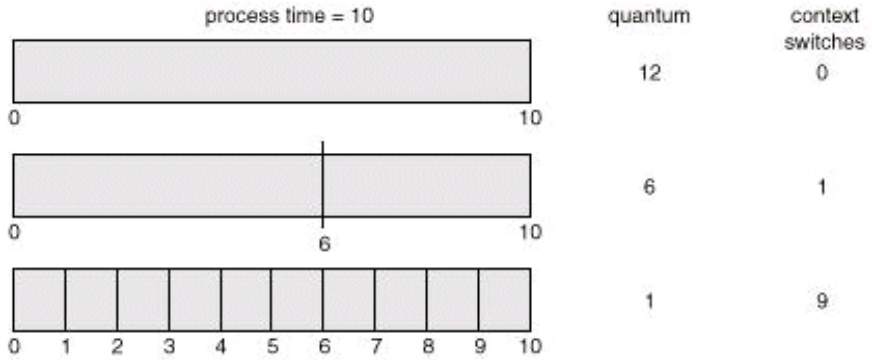
Planowanie priorytetowe

- Z każdym procesem związana jest liczba całkowita określająca jego priorytet.
- Procesor przydzielany jest procesowi z najwyższym priorytetem (zwykle mniejsza liczba to większy priorytet).
- Dwa warianty: z wywłaszczaniem i bez.
- SJF można traktować jako priorytetową, gdzie długość fazy określa priorytet.
- Możliwość zagłodzenia — procesy z niskim priorytetem zawsze są spychane na koniec kolejki.
- Postarzanie — priorytet wzrasta w miarę oczekiwania.

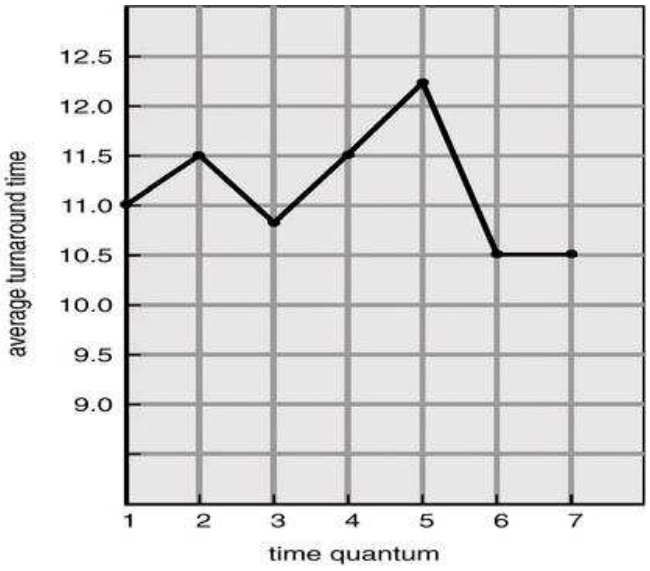
RR: planowanie rotacyjne

- Każdy proces otrzymuje mały odcinek czasu CPU (kwant, rzędu 10-100ms). Po jego wykorzystaniu proces jest wywłaszczany i wędruje na koniec kolejki procesów gotowych.

- Przy n procesach i kwancie q każdy proces dostaje $1/n$ czasu w odcinkach co najwyżej wielkości q. Każdy proces czeka nie dłużej niż $q(1-n)$ jednostek czasu.
- Zachowanie:
  - q duże, $\rightarrow$ FIFO;
  - q małe, $\rightarrow$ jeśli zbyt małe w stosunku do czasu przełączania kontekstu, narzut staje się zbyt duży.



Typowo średni czas cyklu przetwarzania większy niż dla SJF, ale lepszy czas reakcji.



process	time
P_1	6
P_2	3
P_3	1
P_4	7

Kolejki wielopoziomowe

Kolejka procesów gotowych w naturalny sposób da się podzielić na kilka oddzielnych kolejek, np.:

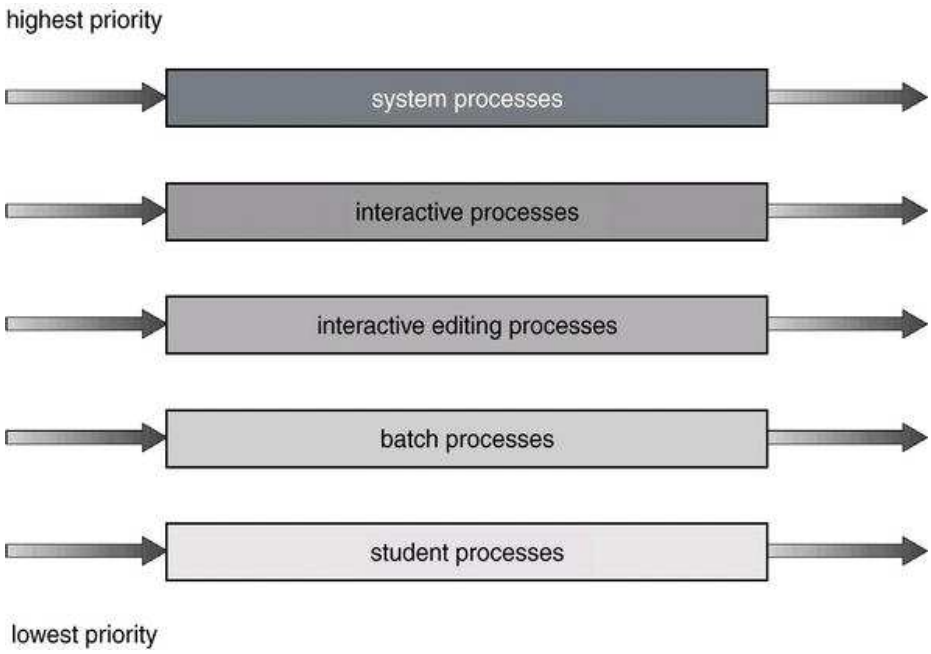
- procesy pierwszoplanowe (interakcyjne),
- drugoplanowe (wsadowe).

Każda z kolejek ma swój algorytm planowania, np.:

- pierwszoplanowe — RR,
- drugoplanowe — FCFS.

Konieczne jest planowanie(szeregowanie) pomiędzy kolejkami, np.:

- stałe priorytety poszczególnych kolejek — możliwe zagłodzenie,
- każda kolejka dostaje część czasu do dyspozycji dla swoich procesów, np.: 80% dla procesów pierwszoplanowych, 20% dla drugoplanowych.

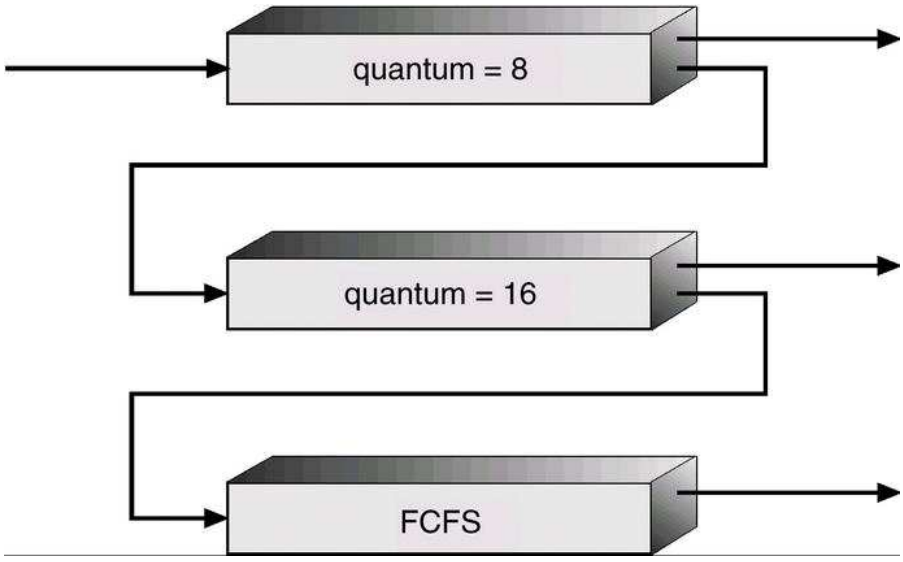


Kolejki wielopoziomowe ze sprzężeniem zwrotnym

Procesy mogą być przemieszczane z jednej kolejki do drugiej — zarówno o niższym (za kare, np. zużywa zbyt dużo procesora) jak i o wyższym (postarzanie) priorytecie.

Parametry kolejki:

- liczba kolejek,
- metoda planowania dla każdej kolejki,
- kryteria awansu do kolejki o wyższym priorytecie,
- kryteria degradacji do kolejki o niższym priorytecie,
- metoda wyboru kolejki początkowej dla procesu.



Przykład:

Proces trafia do kolejki Q_0 i otrzymuje kwant czasu wielkości 8 jednostek. Jeśli wykorzysta cały kwant zostaje przeniesiony do kolejki Q_1. Jeśli nie pozostaje w swojej kolejce.

Analogicznie w kolejce Q_1 — wykorzystanie kwantu powoduje degradację.

Kolejki obsługiwane są zgodnie z priorytetami.

Stosowany jest mechanizm wywłaszczania.

Proces nie może awansować do kolejki o wyższym priorytecie.

Planowanie dla wielu procesorów

Bardziej skomplikowane, żadna z metod nie okazała się lepsza niż inne.

Systemy homogeniczne — wszystkie procesory identyczne.

Dzielenie obciążeń — wspólna kolejka procesów gotowych.

- Każdy procesor sam planuje swoje działanie — wymaga starannej ochrony struktur danych (procesy nie mogą ginąć, każdy może zostać wybrany tylko przez jeden procesor, itp).
- Schemat master-slave pozwala na uniknięcie tych problemów.

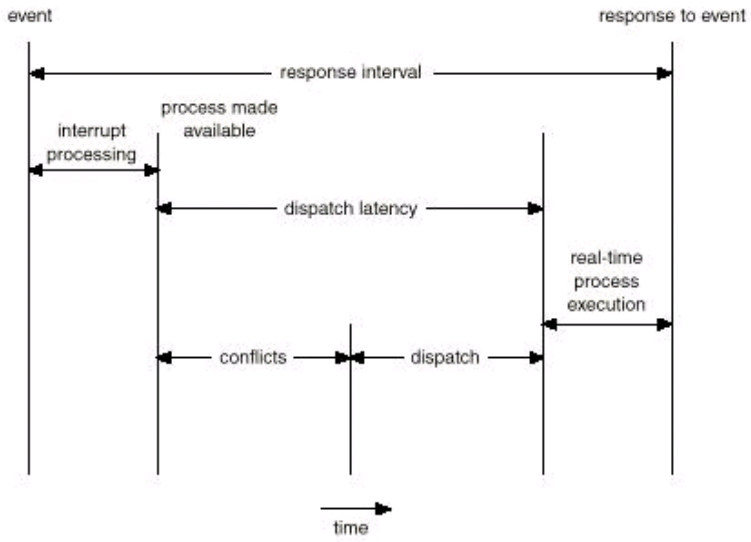
Planowanie w czasie rzeczywistym

Dwa typy systemów:

- rygorystyczne — krytyczne zadanie musi być wykonane w określonym czasie,
- łagodne — krytyczne procesy mają wyższy priorytet.

Łagodny tryb przetwarzania w czasie rzeczywistym może być wbudowany do systemu ogólnego przeznaczenia — grafika, multimedia.

- Priorytety procesów czasu rzeczywistego nie mogą maleć.
- Opóźnienie ekspedycji musi być małe — wywłaszczanie funkcji systemowych (punkty wywłaszczeń), wywłaszczanie jądra.



Faza konfliktu obejmuje:

- wywłaszczenie procesu działającego w jądrze,
- zwolnienie przez niskopriorytetowe procesy zasobów potrzebnych procesowi wysokopriorytetowemu,
- przełączenie kontekstu.

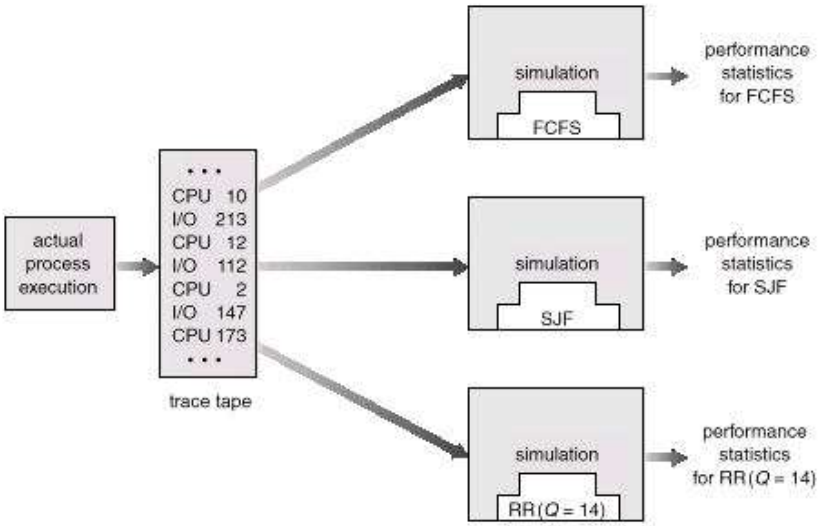
Ocena algorytmów

Jak wybrać algorytm planowania w konkretnym przypadku?

Ustalenie kryteriów.

- Modelowanie deterministyczne — jedna z metod oceny analitycznej. Przyjmuje się konkretne obciążenie systemu i definiuje zachowanie algorytmów w systemie. Wymaga dokładnych danych wejściowych.
- Modele obsługi kolejek. Mierzymy rozkłady faz procesora, rozkłady przybycia zadań, itp. Na tej podstawie obliczamy np. średnie czasy oczekiwania w poszczególnych kolejkach.

- Symulacje. Mogą być kosztowne. Sensowne projektowanie i realizacja nie są łatwe.
- Implementacja. Koszt. Zmieniające się środowisko.



Planowanie w systemie Solaris2