PROCESY I WĄTKI W SYSTEMACH ROZPROSZONYCH

Proces z jednym wątkiem sterowania:
własny licznik rozkazów, stos, zbiór rejestrów, przestrzeń adresowa;
komunikacja między procesami - systemowe mechanizmy komunikacji np. komunikaty (ew. semafory ale tylko wtedy gdy dostępna jest pamięć współdzielona).

Wiele wątków sterowania w procesie:
każdy wątek ma własny licznik rozkazów, stos, rejestry;
ale wszystkie wątki maj ą wspólną przestrzeń adresową, ten sam zbiór otwartych. plików, procesów pochodnych itp.

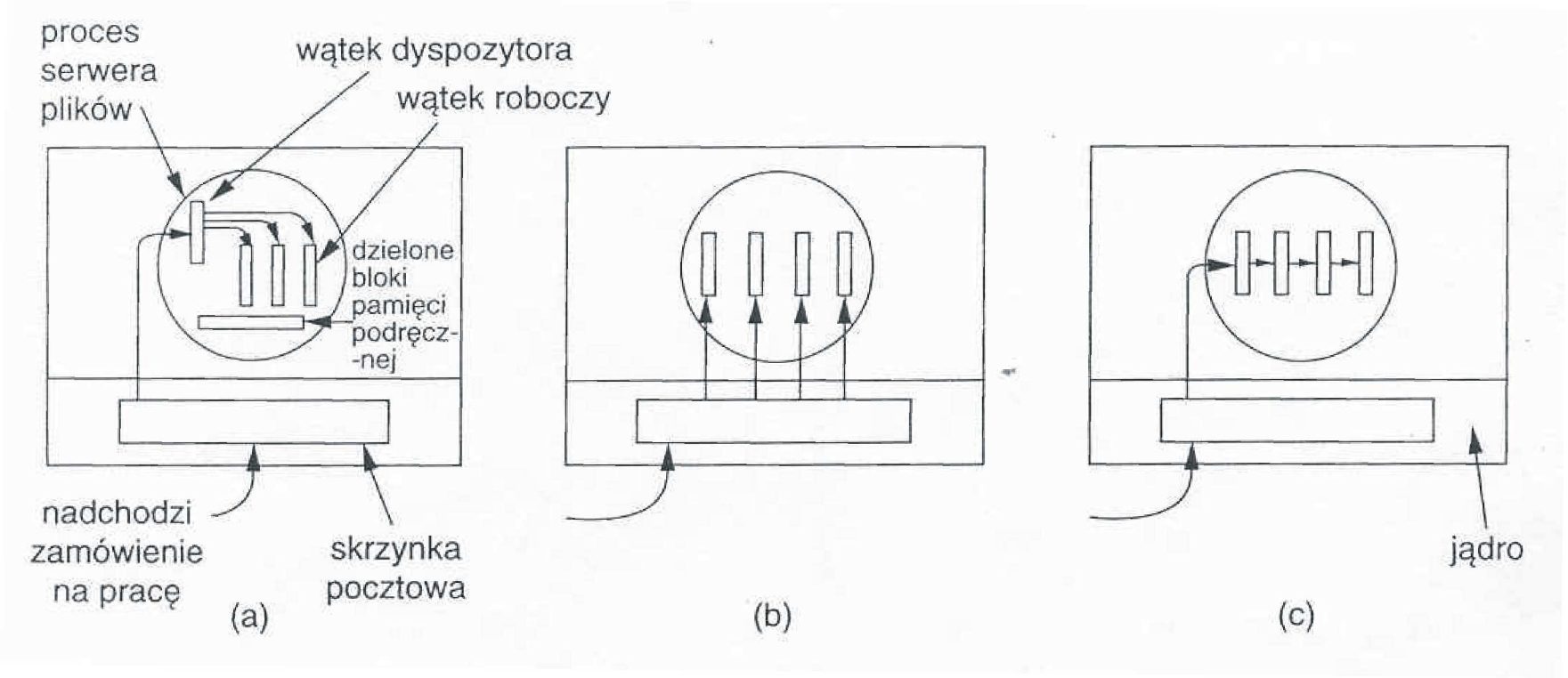
Cechy analogiczne (wątków i tradycyjnych procesów jednowątkowych) :
wykonywanie sekwencyjne, działanie współbieżne - podział czasu procesora,
stany wątków: gotowy, wykonywany, czekający (blokowany), likwidowany;
tworzenie wątków pochodnych.

Tabela. Elementy procesu (w którego skład wchodzi wiele wątków) wspólne dla wszystkich wątków oraz elementy odrębne dla każdego pojedynczego wątku.

Proces	**Wątek**
przestrzeń adresowa	licznik rozkazów
zmienne globalne	stos
otwarte pliki	zbiór rejestrów
procesy pochodne	wątki pochodne
czasomierze	stan
sygnały	
semafory	
informacje rozrachunkowe	

Sposoby organizacji wątków w procesie

Dyspozytor - pracownik (Dispatcher – worker)
Model zepołowy (Team model)
Model potokowy (Pipeline model)



Na rysunku trzy sposoby organizacji wątków w procesie:
(a) dyspozytor-pracownik, (b) model zespołowy, (c) model potokowy

Pakiety wątków

Pakiet wątków: zbiór elementarnych działań - wywołań bibliotecznych dostępnych dla programistów.

Przykłady działań elementarnych:

tworzenie nowego wątku
likwidacja wątku
czekanie na zakończenie innego wątku
informowanie planisty, że w tym momencie powinien być wykonany inny wątek.

Organizacja wzajemnego wyłączania

zastosowanie wirujących blokad,
elementów synchronizacji:
zamek (mutex) - zmienna do wyłącznego dostępu,
typowe operacje:

```
lock(zamek);
unlock(zamek);
```

zmienna warunkowa
typowe operacje:

```
wait(zmienna warunkowa);
wakeup( zmienna warunkowa);
```

Przykład: zapewnienie wyłączności dostępu do zasobu:

```
lock(zamek);
        sprawdź struktury danych;
        while(zasób zajęty)
                wait(zmienna warunkowa);
        oznacz zasób jako zajęty;
unlock(zamek);
. . . . . .
korzystanie z zasobu
. . . . . .
lock(zamek);
        oznacz zasób jako wolny;
unlock(zamek);
wakeup(zmienna warunkowa);
```

Problemy planowania wątków na przykładzie systemu Mach

Założenia i cele projektu Mach

System operacyjny przeznaczony do pracy w systemach rozproszonych, zgodny z systemem BSD UNIX.
Możliwość pracy w systemach heterogenicznych.
Możliwość pracy w systemach komputerowych o różnej architekturze sprzętowej, (w tym z wieloprocesorami).
Możliwość pracy w sieciach komputerowych o różnej prędkości.
Zapewnienie klientom przezroczystości sieci i obiektowej organizacji.
Zintegrowane zarządzanie pamięcią i komunikacją międzyprocesową.

Pojęcie wielowątkowego procesu - zadania (task) i wątku.

Charakterystyka planowania w systemie Mach

Problem planowania: wiele procesów-zadań, wiele wątków, wiele procesorów.
Planuje się tylko przydział procesorów do wątków.
System priorytetów przypisanych wątkom.
Kolejki globalne wykonywanych wątków
Kolejki lokalne przypisane procesorom.
Rozproszona koordynacja przydziału wątków do procesorów.
Zmienny kwant czasu w systemie.

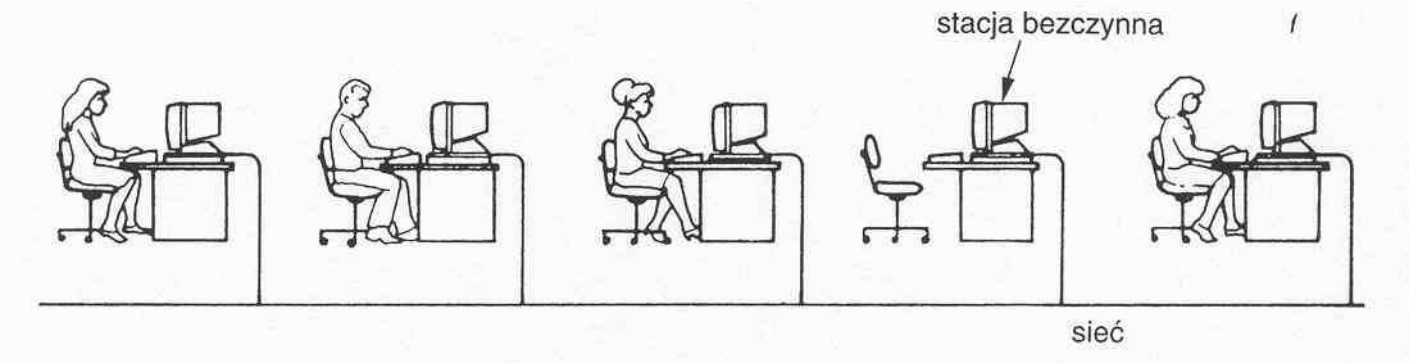
PROCESY I WĄTKI W SYSTEMACH ROZPROSZONYCH

MODELE SYSTEMÓW

Różne sposoby organizacji systemów

Model stacji roboczych

Wiele stacji połączonych siecią LAN



Rys. 4.10. Sieć osobistych stacji roboczych wyposażonych w lokalne systemy plików

Możliwe wykorzystanie stacji z prywatnymi dyskami, ale także stacji bezdyskowych.
Stacje bezdyskowe - systemy plików realizowane na zdalnych serwerach.
Zalety stacji bezdyskowych:

- niskie koszty,
- łatwość eksploatacji,
- symetria wykorzystania,
- niski hałas.

Sposoby wykorzystania prywatnych dysków stacji roboczych

Stronicowanie i przechowywanie plików tymczasowych

pliki tymczasowe, tworzone w czasie sesji np. w trakcie kompilacji, nie muszą być przesyłane do serwera plików.

Stronicowanie i przechowywanie plików tymczasowych, oraz systemowych plików binarnych

na dyskach lokalnych przechowuje się dodatkowo najczęściej wykorzystywane binaria - kompilatorów, edytorów tekstu, programy obsługi.

Stronicowanie i przechowywanie plików tymczasowych, systemowych plików binarnych, oraz podręczna pamięć plików

w czasie sesji użytkownik ściąga potrzebne pliki z serwera na dysk lokalny, pracuje wykorzystując dysk lokalny, odsyła ostateczne wersje plików do serwera przed zakończeniem sesji.
Zalety: redukcja obciążenia sieci, utrzymuje się zcentralizowaną pamięć długoterminową.
Wady: Problem utrzymania spójności pamięci podręcznych.

Kompletny lokalny system plików
Każda maszyna ma własny system plików z możliwością montowania systemów plików innych maszyn.

Zalety:
gwarantowany czas odpowiedzi,
małe obciążenie sieci.

Wady:
utrudnione dzielenie informacji,
realizuje idee operacyjne system sieciowy, a nie przezroczystego systemu rozproszonego.

Wykorzystanie bezczynnych stacji

Ogólny problem zdalnego wykonywania procesów w sposób przezroczysty.

Pierwsza próba - UNIX BSD

```
rsh maszyna polecenie
```

wady: trzeba określić maszynę, środowisko zdalne na ogół inne niż lokalne.

Problemy

znalezienie bezczynnej maszyny,
zapewnienie przezroczystości wykonania,
czynności po powrocie właściciela.

Znalezienie bezczynnej stacji

Definicja bezczynności stacji.

Algorytm lokalizacji bezczynnej stacji sterowany za pomocą serwera

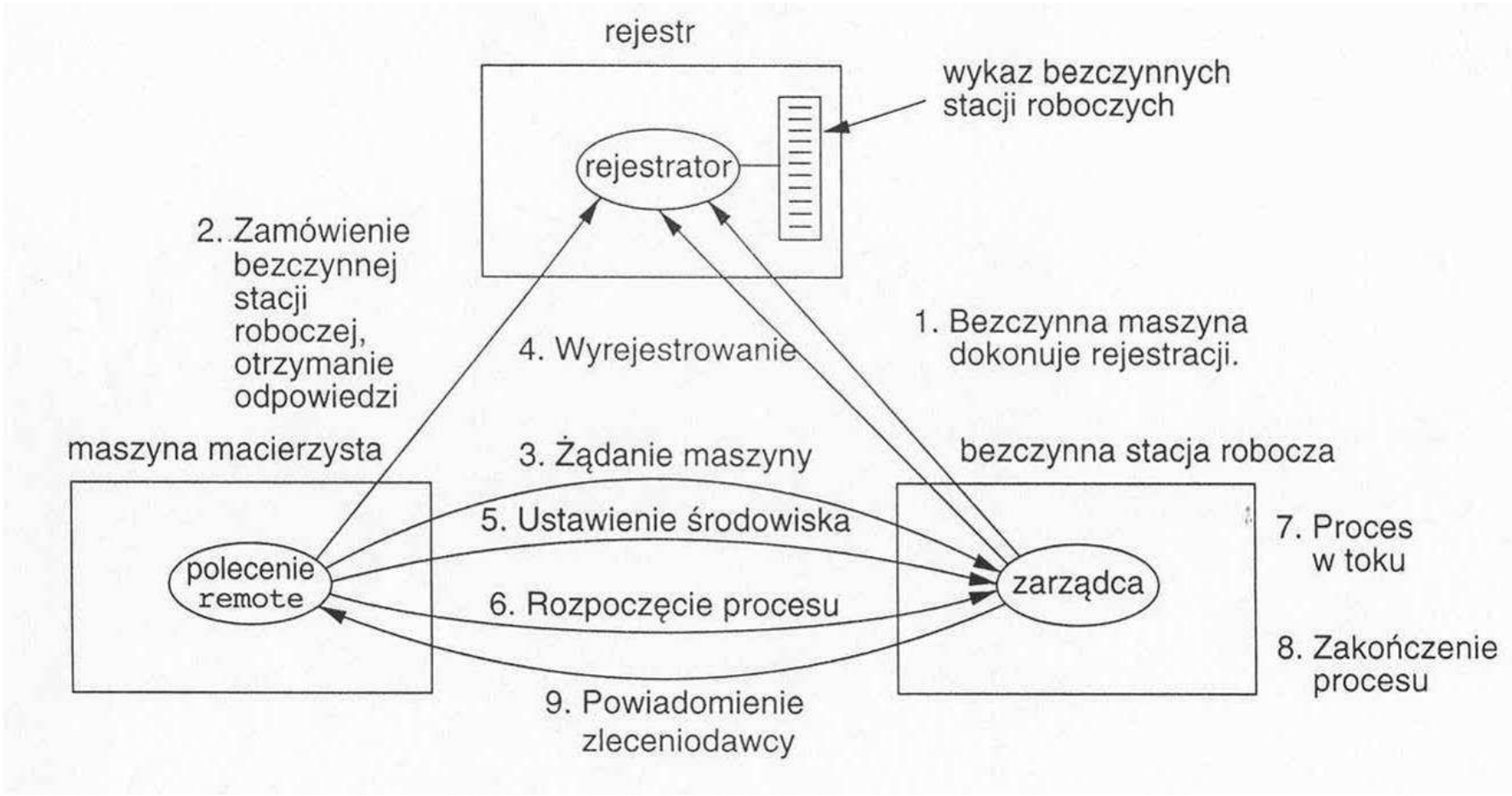
Stacja robocza

stwierdza swoją bezczynność
ogłasza swoją dostępność - niezbędne informacje (dane stacji) są wpisywane do pliku rejestracyjnego

Użytkownik

wykonuje: `remote polecenie`, program `remote` sam sprawdza rejestr

Algorytm znajdowania i zatrudniania bezczynnej stacji roboczej wykorzystujący centralne rejestrowanie



Algorytm sterowany przez klienta

Program `remote` rozgłasza zamówienie, podaje jako informacje:
- program, który potrzebuje stację,
- wielkość potrzebnej pamięci,
- zapotrzebowanie na obliczenia, ...

Po nadejściu odpowiedzi program `remote` wybiera stację.

Zdalne wykonanie procesu:
- przemieszczenie kodu,
- zapewnienie tego samego środowiska
  - potrzebny ten sam obraz plików
  - ten sam katalog roboczy
  - te same zmienne środowiska.

Problemy działania jądra systemu

Odwołania do systemu plików np. operacja read:
- system bezdyskowy - zamówienie do serwera plików
- dyski lokalne z kompletnymi systemami plików - do stacji macierzystej.

Odwołania dot. klawiatury i monitora: przesyłane do stacji macierzystej.

Inne odwołania, np. dot. priorytetu, segmentu danych, nazwy maszyny, adresu sieciowego itp.: wykonywane zdalnie.

Problemy synchronizacji czasu.

Powrót właściciela do stacji

Jakie podjąć działania

Żadne - stacja przestaje być osobista

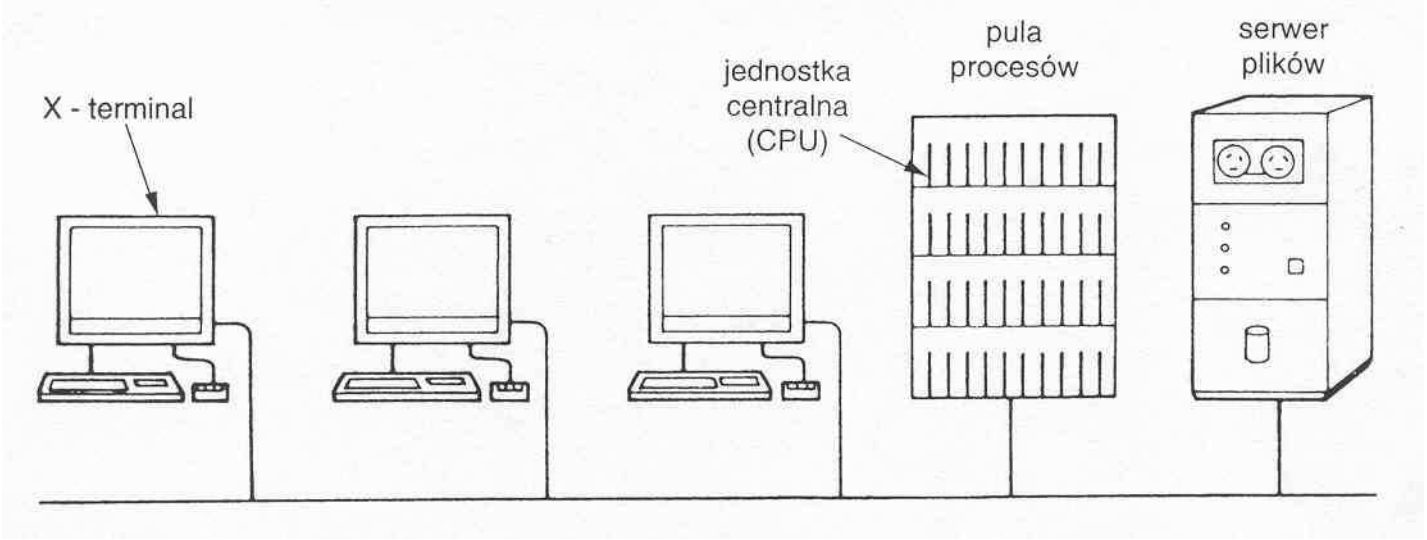
Zlikwidować proces zewnętrzny

nagle - utrata całej pracy, system w chaosie
ostrzec proces, aby mógł sam się zamknąć
przenieść proces na inną maszynę
tzn. kod i dane użytkownika, jądrowe struktury danych

Oczyścić maszynę źródłową.

Podstawowa zasada:
kończący proces musi zostawić maszynę w takim samym stanie, w jakim ją zastał.

Model puli procesów
Wiele jednostek centralnych w jednej szafie.
Użytkownicy mają szybkie terminale graficzne.

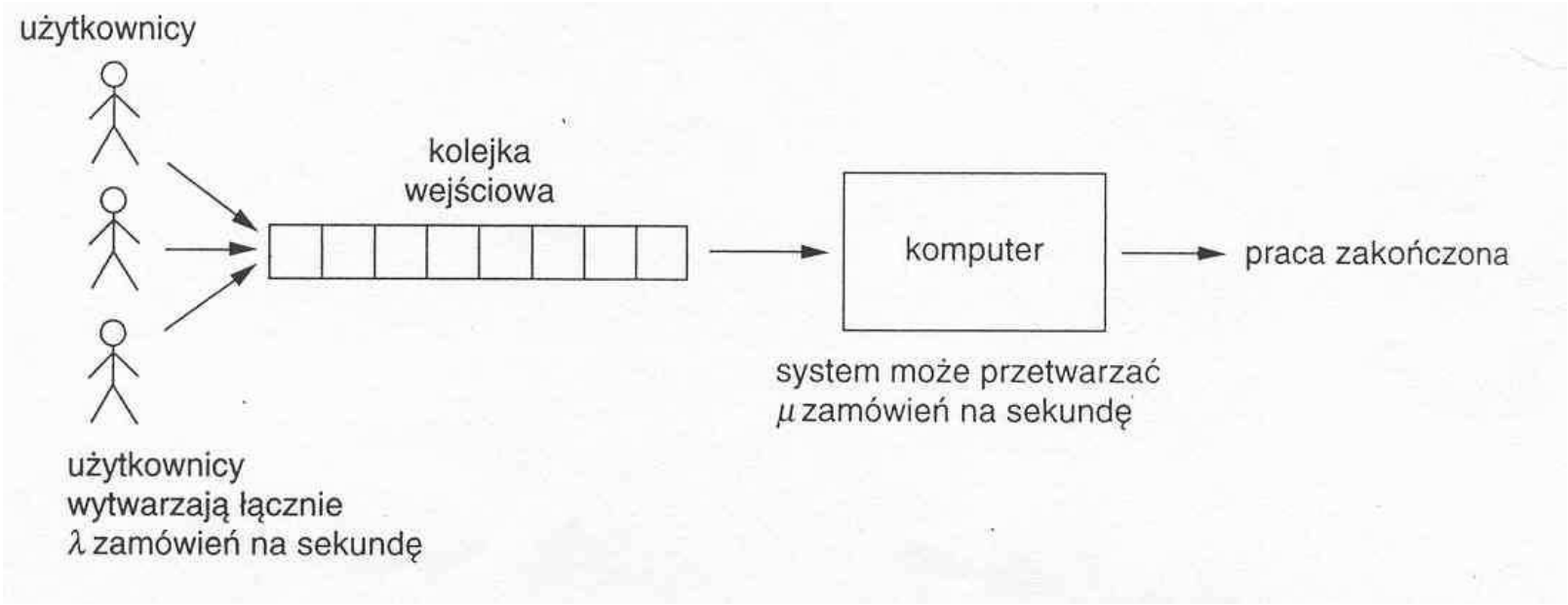


Rys. Przykład systemu rozproszonego wg. modelu puli procesorów

Zalety:
redukcja kosztów - wspólny system zasilania, obudowa, ...
łatwość powiększania mocy obliczeniowej,
możliwość udostępnienia użytkownikowi tylu procesorów, ile potrzebuje.

System masowej obsługi

użytkownicy generują losowo zamówienia,
zamówienia ustawiane są w kolejce do obsługi.



Rys. Elementarny system masowej obsługi