



Synchronizacja cz. IV

Łączy mogą gwarantować również, w sposób niewidoczny dla użytkownika, że żadna wiadomość nie jest tracona, duplikowana lub zmieniana - są to tzw. **łączy niezawodne** (ang. reliable, lossless, duplicate free, error free, uncorrupted, no spurious).

Czas transmisji w łączy niezawodnym (ang. transmission delay, in-transit time) może być ograniczony lub jedynie określony jako skończony lecz nieprzewidywalny. W pierwszym przypadku mówimy o **transmisji synchronicznej** lub z **czasem deterministycznie ograniczonym** (w szczególności równym zero), a w drugim - o **transmisji asynchronicznej** lub z **czasem niedeterministycznym**.

Operacje wymiany komunikatów

Wymiana komunikatów (ang. message passing) realizowana jest z użyciem dwóch podstawowych operacji komunikacyjnych:

- ❖ $\text{send}(P, m)$
- ❖ $\text{receive}(Q, m)$

gdzie: m jest przesyłanym komunikatem (wiadomością, ang. message),
 P - jest odbiorcą komunikatu,
 Q - jest nadawcą komunikatu.

Określanie nadawców i odbiorców komunikacja bezpośrednia

Procesy mogą komunikować się bezpośrednio lub pośrednio. W komunikacji bezpośredniej każdy proces, który chce nadać lub odebrać komunikat musi jawnie nazwać odbiorcę lub nadawcę uczestniczącego w tej wymianie informacji. W tym wypadku operacje send i receive są zdefiniowane następująco:

- ❑ $\text{send}(P, m)$ - nadaj komunikat m do procesu P
- ❑ $\text{receive}(Q, m)$ - odbierz komunikat od procesu Q

Łączy komunikacyjne mają tu następujące własności:

- ❖ ustawiane są automatycznie między parą procesów, które mają komunikować się;
- ❖ dotyczą dokładnie dwóch procesów;
- ❖ są dwukierunkowe.

Przedstawiony schemat charakteryzuje się symetrią adresowania



Łączy komunikacyjne jest elementem umożliwiającym transmisję informacji między interfejsami odległych węzłów. Wyróżnia się łączy jedno i dwukierunkowe. Wyposażone są one w *bufory* o określonej *pojemności* (ang. links capacity).

Jeżeli łączy nie posiada buforów (jego pojemność jest równa zero), to mówimy o **łączy niebuforowanym**, w przeciwnym razie - o **buforowanym**.

Zwykle kolejność odbierania komunikatów wysyłanych z danego węzła jest zgodna z kolejnością ich wysłania, wówczas łączy nazywamy **łączy FIFO**, w przeciwnym razie - **nonFIFO**.

Istnieje też asymetryczny wariant adresowania, w którym nadawca nazywa odbiorcę, a od odbiorcy nie wymaga się znajomości nadawcy. W tym wypadku operacje send i receive są zdefiniowane następująco:

- ❖ $\text{send}(P, m)$ - nadaj komunikat m do procesu P ;
- ❖ $\text{receive}(id, m)$ - odbierz komunikat od dowolnego procesu; pod id zostanie podstawiona nazwa procesu, od którego nadszedł komunikat.



Określanie nadawców i odbiorców komunikacja pośrednia

W komunikacji pośredniej komunikaty są nadawane i odbierane poprzez skrzynki pocztowe (nazywane też portami, ang. mailbox).

Abstrakcyjna skrzynka pocztowa jest obiektem, w którym procesy mogą umieszczać komunikaty, i z którego komunikaty mogą być pobierane. Każda skrzynka pocztowa ma jednoznaczny identyfikator. Proces może komunikować się z innymi procesami za pomocą różnych skrzynek pocztowych. W tym wypadku operacje *send* i *receive* są zdefiniowane następująco:

- *send* (*A*, *m*) – nadaj komunikat *m* do skrzynki *A*
- *receive* (*A*, *m*) – odbierz komunikat ze skrzynki *A*

Łąca komunikacyjne mają tu następujące własności:

- ❖ ustawiane są między procesami tylko wówczas, gdy procesy te dzielą jakąś skrzynkę pocztową
- ❖ mogą wiązać więcej niż dwa procesy
- ❖ każda para procesów może mieć kilka różnych łączy
- ❖ mogą być jednokierunkowe lub dwukierunkowe.



Slajd 8

(c) Zakład Systemów Informatycznych

Producent – Konsument

```
1. program PRODUCERCONSUMER_MESSAGETRANSMISSION;
2. var bufferPool: array[0..X] of BUFFER;
3. procedure PRODUCER;
4. begin
5.   while True do
6.     begin
7.       produceNextMessage;
8.       receive(producer, empty); /* odbiór blokowany */
9.       addMessageToCommonBuffer;
10.      send(consumer, empty); /* wysłanie asynchroniczne */
11.    end;
12.  end;
13. procedure CONSUMER;
14. begin
15.   while True do
16.     begin
17.       receive(consumer, empty);
18.       takeMessageFromCommonBuffer;
19.       send(producer, empty);
20.       processMessage;
21.     end;
22.   end;
23. begin
24.   I:=N;
25.   while I>0 do
26.     begin
27.       send(producer, empty);
28.       I:=I-1;
29.     end;
30.   parbegin
31.     PRODUCER;
32.     CONSUMER;
33.   parend
34. end.
```



Slajd 11

(c) Zakład Systemów Informatycznych

Skrzynka pocztowa

Skrzynka może być własnością procesu lub systemu. Jeżeli skrzynka należy do procesu (tzn. jest przypisana lub zdefiniowana jako część procesu), to różni się jej *właściciela* (który za jej pośrednictwem może tylko odbierać komunikaty) i *użytkownika* (który może tylko nadawać komunikaty do danej skrzynki).

W wielu przypadkach, proces ma możliwość zadeklarowania *zmiennej typu skrzynka pocztowa*. Proces deklarujący skrzynkę pocztową staje się jej właścicielem. Każdy inny proces, który zna nazwę tej skrzynki, może zostać jej użytkownikiem.

Skrzynka pocztowa należąca do systemu istnieje bez inicjatywy procesu i dlatego jest niezależna od jakiegokolwiek procesu. System operacyjny dostarcza mechanizmów pozwalających na:

- ✓ tworzenie nowej skrzynki;
- ✓ nadawanie i odbieranie komunikatów za pośrednictwem skrzynki;
- ✓ likwidowanie skrzynki.

Proces, na którego zamówienie jest tworzona skrzynka, staje się domyślnie jej właścicielem. Przywilej własności jak i odbierania komunikatów może jednak zostać przekazany innym procesom za pomocą odpowiednich funkcji systemowych.



Slajd 9

(c) Zakład Systemów Informatycznych

Operacje synchroniczne i asynchroniczne

Kanały o niezerowej pojemności umożliwiają realizację następujących operacji komunikacji:

❖ *nieblokowanych (asynchronicznych)*

proces nadający przekazuje komunikat do kanału (bufora) i natychmiast kontynuuje swe działanie, a proces odbierający odczytuje stan kanału wejściowego, lecz nawet gdy kanał jest pusty, proces kontynuuje działanie;

❖ *blokowanych (synchronicznych)*

nadawca jest wstrzymywany do momentu, gdy wiadomość zostanie odebrana przez adresata, natomiast odbiorca - do momentu, gdy oczekiwana wiadomość pojawi się w jego buforze wejściowym.

W komunikacji *synchronicznej*, nadawca i odbiorca są blokowani aż odpowiedni odbiorca odczyta przesłaną do niego wiadomość (ang. rendez-vous). W przypadku komunikacji *asynchronicznej*, nadawca lub odbiorca komunikuje się w sposób nieblokowany.



Slajd 10

(c) Zakład Systemów Informatycznych