

## Wywołanie procedur zdalnych

---



---



---



---



---



---



---

## Mechanizm wywołania procedury

```

main(int argc, char* argv){
    int id, status;
    id = atoi(argv[1]);
    status = zabij_proc(id);
    exit(status)
}
.
.
.
int zabij_proces (int pid){
    int stat;
    stat = kill(pid, 9);
    return stat;
}

```

---



---



---



---



---

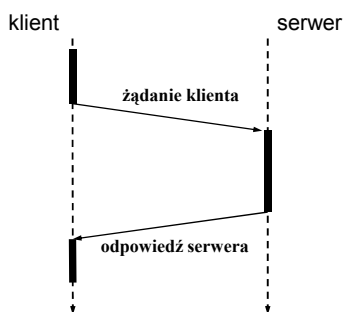


---



---

## Schemat interakcji klient-serwer (transakcja komunikatu)




---



---



---



---



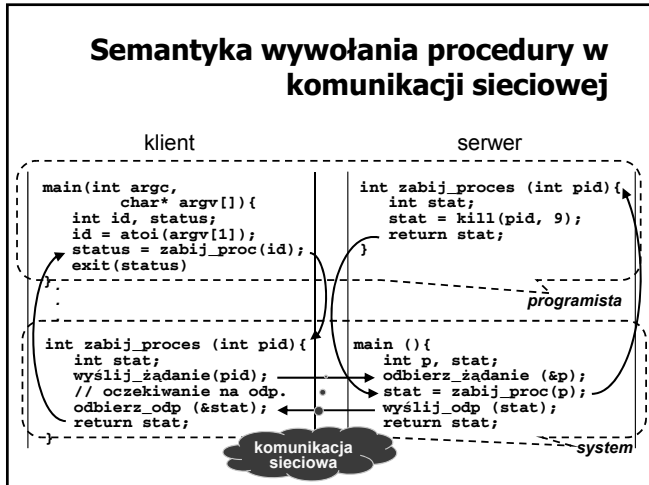
---



---



---




---

---

---

---

---

---

---

---

---

---

### RPC — zagadnienia projektowe

- ☞ Przezroczystość dostępu — ukrycie komunikacji sieciowej przed aplikacją przez odpowiednie opakowanie funkcji komunikacyjnych namiastką klienta oraz serwera.
- ☞ Gwarancja wykonania — ukrywanie błędów komunikacyjnych
- ☞ Specyfikacja interfejsu — sposób opisu sygnatur procedur zdalnych (nazwy, typy parametrów)
- ☞ Obsługa sytuacji wyjątkowych

---

---

---

---

---

---

---

---

---

---

### RPC — przezroczystość dostępu

- ☞ Namiastka klienta (ang. client stub) — udostępnienie aplikacji klienckiej procedury lokalnej odpowiedzialnej za przesłanie danych do serwera oraz odebranie wyników
- ☞ Namiastka serwera (ang. server stub) — udostępnienie aplikacji po stronie serwera procedury lokalnej odpowiedzialnej za odebranie identyfikatora procedury zdalnej do wywołania, parametrów procedury, a odesłanie wyników lub zgłoszenie wyjątków

---

---

---

---

---

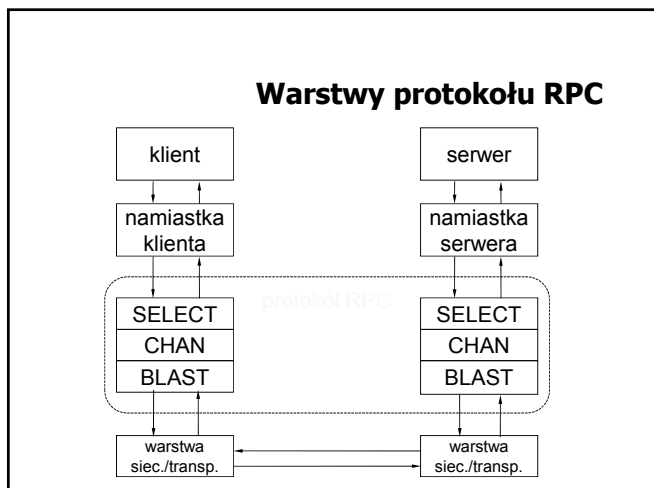
---

---

---

---

---




---

---

---

---

---

---

---

---

- ### Interakcja klient-serwer w realizacji wywołania zdalnego
1. Lokalne wywołanie procedury namiastki przez klienta
  2. Przygotowanie (upakowanie) danych do wysłania na stronę serwera
  3. Wysłanie przygotowanego komunikatu na stronę serwera
  4. Odebranie komunikatu przez serwer
  5. Rozpakowanie danych
  6. Wykonanie procedury zdalnej
  7. Przygotowanie (upakowanie) danych z odpowiedzią dla klienta
  8. Wysłanie przygotowanego komunikatu na stronę klienta
  9. Odebranie komunikatu przez namiastkę klienta
  10. Rozpakowanie komunikatu i zwrócenie klientowi wyniku

---

---

---

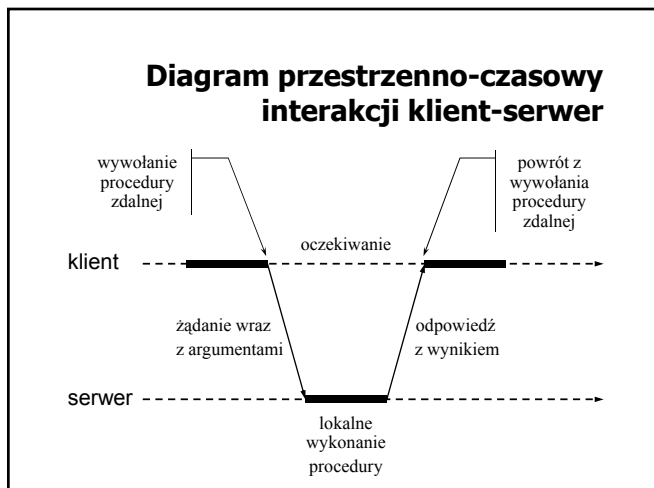
---

---

---

---

---




---

---

---

---

---

---

---

---

### RPC - przekazywanie parametrów

- ☞ przekazywanie przez wartość (ang. call-by-value) — problem reprezentacji danych (np. różnice w kodowaniu znaków, różnice w kolejności bajtów, formaty liczb zmiennopozycyjnych)
- ☞ przekazywanie przez referencje (ang. call-by-reference) — problem zinterpretowania wartości wskaźnika w innej przestrzeni adresowej
- ☞ przekazywanie przez kopiowanie i odtwarzanie (ang. call-by-copy/restore) — utworzenie kopii parametru po stronie procedury i zapis dokonanych tam zmian w procesie klienta po jej zakończeniu (problem opisu struktur danych w celu prawidłowego zidentyfikowania wszystkich składowych)

---



---



---



---



---

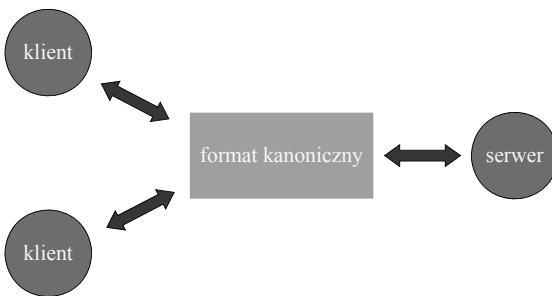


---



---

### Reprezentacja kanoniczna




---



---



---



---



---

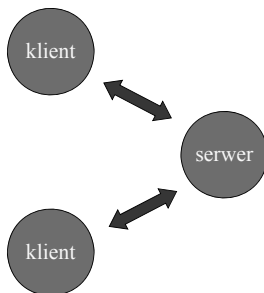


---



---

### Konwersja bezpośrednia




---



---



---



---



---



---



---

### Przekazywanie referencji przez kopiowanie i odtwarzanie

1. Skopiowanie wskazanej wartości do bufora komunikacyjnego i wysłanie do serwera.
2. Wywołanie po stronie serwera procedury zdalnej ze wskaźnikiem na kopię wartości utworzoną po stronie serwera.
3. Skopiowanie zmodyfikowanej wartości z przestrzeni adresowej serwera do bufora komunikacyjnego i przesłanie z powrotem do klienta.
4. Umieszczenie odebranej wartości w miejscu wskazywanym przez referencję po stronie klienta.

---



---



---



---



---

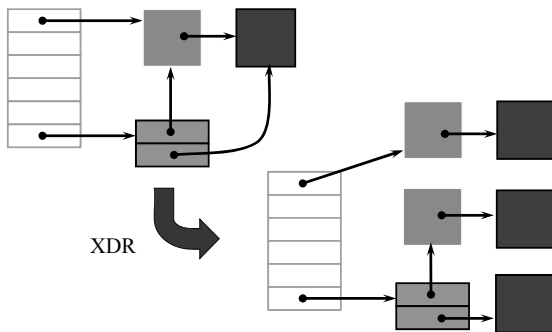


---



---

### Kopiowanie i odtwarzanie — przykład złożonej struktury danych




---



---



---



---



---



---



---

### RPC — gwarancja wykonania

- ☞ Semantyka *ewentualnie* — brak gwarancji, procedura mogła się wykonać lub mogła się nie wykonać.
  - ☞ Semantyka *co najmniej raz* — po uzyskaniu **odpowiedzi z wynikiem** od serwera klient ma pewność, że wywoływana procedura wykonała się co najmniej raz.
  - ☞ Semantyka *co najwyżej raz* — po uzyskaniu **odpowiedzi z wynikiem** od serwera klient wie, że wywoływana procedura wykonała się dokładnie raz.
- Jak należy zinterpretować przypadek wystąpienia błędu (wyjątku) w wywołaniu procedury zdalnej?**
- ☞ Semantyka *dokładnie raz* — niemożliwa do uzyskania, jeśli system narażony jest na awarie (np. serwera lub łączy).

---



---



---



---



---



---



---

### Specyfikacja interfejsu

- ☞ Opis interfejsu w języku implementacji (np. Ada, Java RMI)
- ☞ Opis interfejsu w języku specjalnym, niezależnym od implementacji (CORBA — IDL, Sun RPC — rpcgen)

---

---

---

---

---

---

---

---

### RPC — zagadnienia realizacyjne

- ☞ Przetwarzanie interfejsu
- ☞ Wiązanie klienta z serwerem
- ☞ Obsługa komunikacji klient-serwer
- ☞ Realizacja semantyki błędu
- ☞ Problem osieroconych obliczeń

---

---

---

---

---

---

---

---

### Przetwarzanie interfejsu procedur zdalnych

- ☞ Generowanie namiastki klienta
- ☞ Generowanie namiastki serwera
- ☞ Generowanie przykładowego programu klienta (client sample)
- ☞ Generowanie wzorca do implementacji procedur zdalnych (template)
- ☞ Generowanie plików do zarządzania kompilacją

---

---

---

---

---

---

---

---

### Wiązanie klienta z serwerem

- ☞ Wiązanie statyczne — klient ma na stałe wprowadzony identyfikator komunikacyjny serwera (np. para: adres IP, nr portu).
- ☞ Wiązanie dynamiczne — klient uzyskuje adres serwera za pośrednictwem łącznika (np. portmap, lub rpcbind w Sun RPC).

---

---

---

---

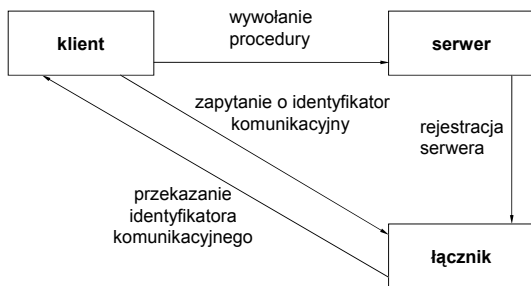
---

---

---

---

### Wiązanie dynamiczne klienta z serwerem




---

---

---

---

---

---

---

---

### Obsługa komunikacji klient-serwer

- ☞ BLAST — realizuje przesyłanie dużych komunikatów poprzez podział na mniejsze części, transmisję poszczególnych części i ponowne złożenie w jeden komunikat po stronie odbiorczej,
- ☞ CHAN — synchronizuje wymianę komunikatów z zadaniami wywołania procedur oraz odpowiedziami,
- ☞ SELECT — rozdziela i przekazuje komunikaty z zadaniami do odpowiednich procesów.

---

---

---

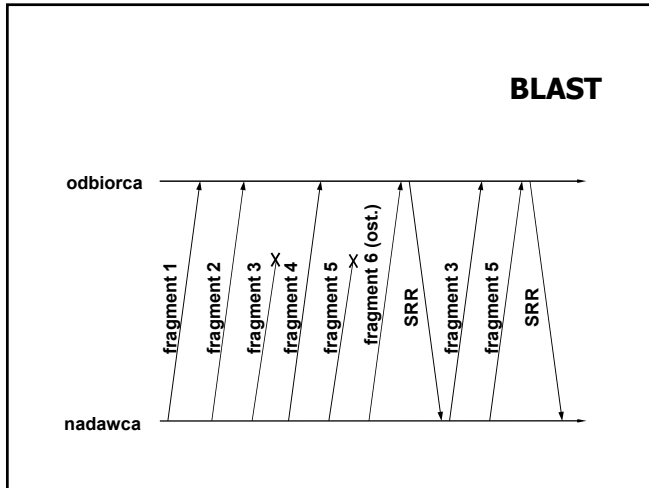
---

---

---

---

---




---

---

---

---

---

---

---

---

---

---

### BLAST — nadawca

1. otrzymanie bloku z w wyższej warstwy stosu protokołów i podział bloku na fragmenty
2. wysłanie kolejno poszczególnych fragmentów do odbiorcy (ze specjalnym oznaczeniem ostatniego fragmentu)
3. ustawienie czasomierza (ang. timer) DONE
4. jeśli dotarł SRR z informacją o brakujących fragmentach, to wysłanie brakujących fragmentów i przejście do pkt. 3
5. jeśli dotarł SRR z informacją o odebraniu wszystkich fragmentów, to SUKCES
6. jeśli DONE = 0 // minął czas oczekiwania to BŁĄD

---

---

---

---

---

---

---

---

---

---

### BLAST — odbiorca (1)

1. po odebraniu pierwszego komunikatu z fragmentem bloku danych:
  - inicjalizacja struktur danych do przechowywania poszczególnych bloków, umieszczenie odebranego fragmentu w odpowiedniej strukturze
  - ustawienie licznika powtórzeń na 0
2. ustawienie czasomierza LAST\_FRAG
3. po odebraniu kolejnego (ale nie ostatniego) komunikatu:
  - umieszczenie nadesłanego fragmentu bloku w odpowiedniej strukturze i przejście do pkt. 2

---

---

---

---

---

---

---

---

---

---



**BLAST — odbiorca (2)**

4. jeśli LAST\_FRAG = 0, to przejście do pkt. 8
5. po odebraniu ostatniego komunikatu umieszczenie nadesłanego fragmentu bloku w odpowiedniej strukturze i sprawdzenie kompletności bloku
6. jeśli blok jest kompletny to wysłanie komunikatu SRR i przekazanie bloku do wyższej warstwy stosu protokołów
7. jeśli blok nie jest kompletny przejście do pkt. 8
8. jeśli licznik powtórzeń jest mniejszy od 3, to wysłanie komunikatu SRR z informacją o brakujących blokach, zwiększenie licznika powtórzeń o 1 i przejście do pkt. 2 w przeciwnym razie rezygnacja z odbioru

---

---

---

---

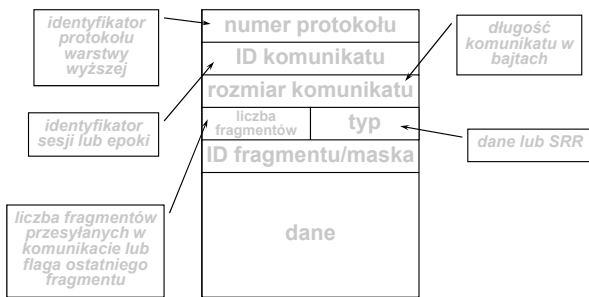
---

---

---

---

**BLAST — format komunikatu**




---

---

---

---

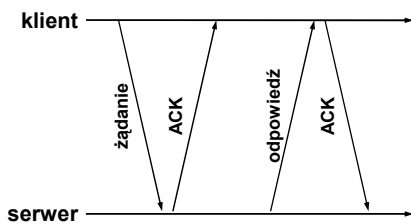
---

---

---

---

**CHAN**




---

---

---

---

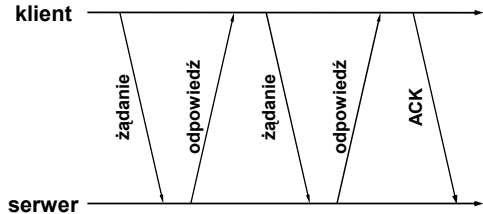
---

---

---

---

**CHAN — domniemane potwierdzenia**




---

---

---

---

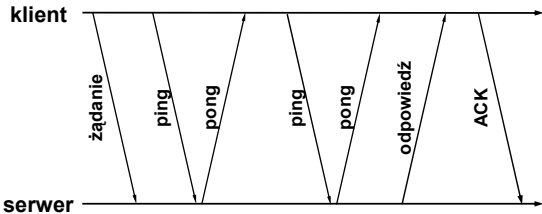
---

---

---

---

**CHAN — próbkowanie serwera**




---

---

---

---

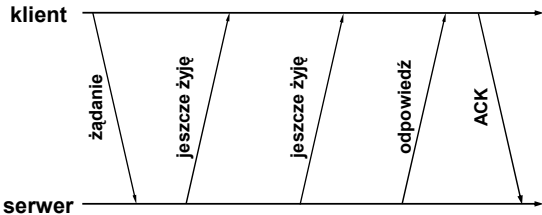
---

---

---

---

**CHAN — „bicie serca”**




---

---

---

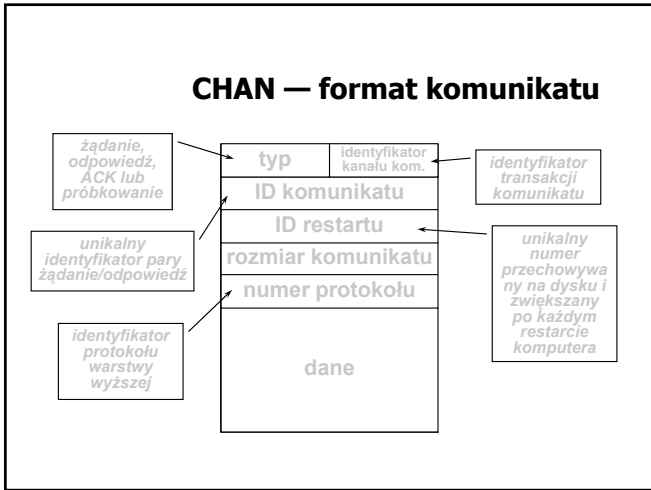
---

---

---

---

---




---

---

---

---

---

---

---

---

- ### Realizacja semantyki błędu
- ☞ nie można zlokalizować serwera — zgłoszenie wyjątku
  - ☞ zaginione żądanie — retransmisja żądania po upływie ustalonego czasu oczekiwania
  - ☞ zaginiona odpowiedź — retransmisja żądania
    - ↳ stosowanie procedur idempotentnych
    - ↳ numerowanie żądań i retransmisja odpowiedzi
  - ☞ awaria serwera
    - ↳ przed podjęciem realizacji → retransmisja żądania
    - ↳ po wykonaniu → zgłoszenie wyjątku
  - ☞ awaria klienta — osierocenie obliczeń

---

---

---

---

---

---

---

---

- ### Usuwanie osieroconych obliczeń (1)
- ☞ Eksterminacja — rejestrowanie działań podejmowanych przez klienta na nośniku niewrażliwym na awarie i usuwanie na tej podstawie osieroconych obliczeń po restarcie klienta.
  - ☞ Reinkarnacja — każdy restart klienta rozpoczyna nową epokę (identyfikowaną przez numer kolejny), po której usuwane są wszystkie obliczenia związane z poprzednią epoką.

---

---

---

---

---

---

---

---

### Usuwanie osieroconych obliczeń (2)

- ☞ Łagodna reinkarnacja — reinkarnacja, w której usuwa się tylko te obliczenia rozpoczęte w starej epoce, dla których nie ma właściciela.
- ☞ Wygaśnięcie — przydział określonego czasu  $T$  serwerowi na wykonanie procedury. Jeśli wykonanie nie zakończy się w czasie  $T$ , serwer musi uzyskać kolejny przydział, pod warunkiem, że obliczenia nie zostały osierocone. Jeśli klient odczeka czas  $T$  przy restarcie, osierocone obliczenia same się zakończą.

---

---

---

---

---

---

---

---

### SELECT

- ☞ Po stronie klienta: odwzorowanie wywoływanej procedury na jej identyfikator, przekazywany do serwera.
- ☞ Po stronie serwera: zlokalizowanie wywoływanej procedury na podstawie identyfikatora.

---

---

---

---

---

---

---

---

### Warianty użycia mechanizmu RPC

- ☞ Wywołanie asynchroniczne
- ☞ Wywołanie zwrotne

---

---

---

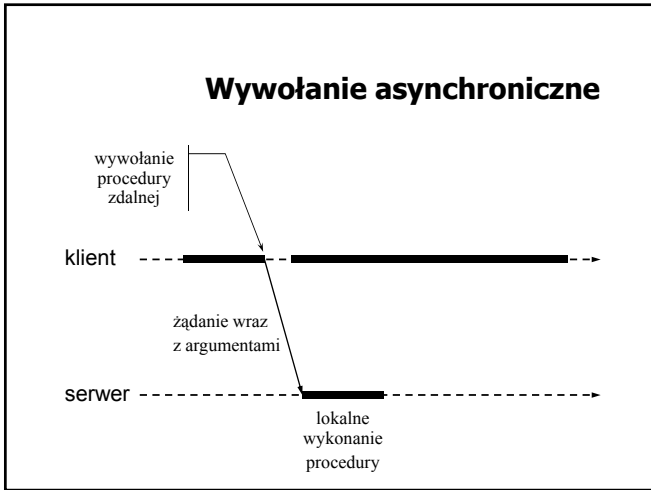
---

---

---

---

---




---

---

---

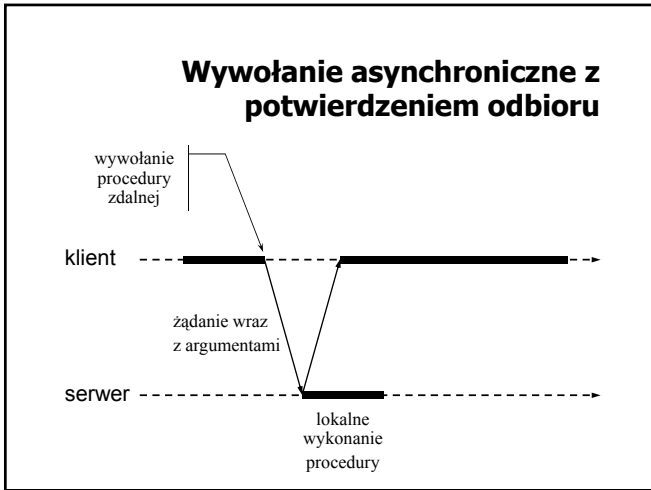
---

---

---

---

---




---

---

---

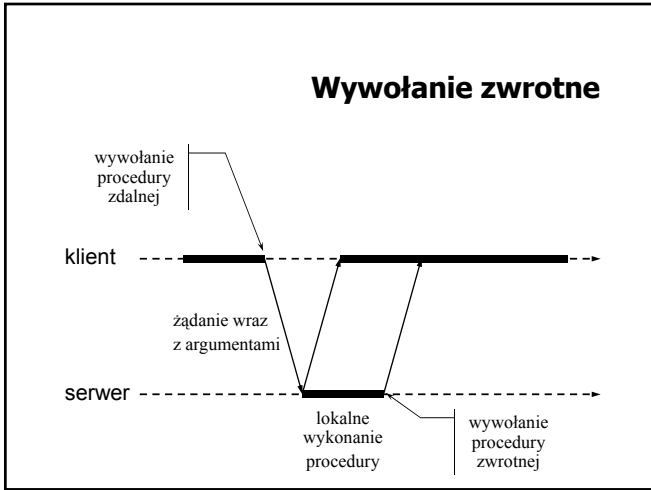
---

---

---

---

---




---

---

---

---

---

---

---

---