# IPC Publish-Subscribe — projekt zaliczeniowy

Protokół komunikacyjny klient-serwer

Filip Ciesielski

Poznań, 13 stycznia 2021

## 1. Informacje ogólne

Stworzony przeze mnie sposób komunikacji, pomiędzy serwerem a klientami logującymi się do systemu, opiera się na jednej kolejce komunikatów, której identyfikujący klucz ma wartość 79630. Poprawna wysyłka żądań działań od klienta do serwera, informacji zwrotnej od serwera do klienta czy przekazywanie wiadomości poszczególnych użytkowników do subskrybujących dany typ innych klientów, możliwa jest dzięki operowaniu na unikalnych typach przesyłanych komunikatów przez wyżej wspomnianą kolejkę. Dodatkowo wprowadzone zostało ograniczenie w ilości użytkowników, którzy mogą zalogować się w systemie. Określa to wartość MAX_NUMBER_OF_USERS i domyślnie ustawiona jest na liczbę 5 klientów. MAX_LOGIN_ATTEMPTS określa maksymalną ilość prób logowania jaką klient może podjąć przed wyrzuceniem go z systemu, domyślnie możliwe są 3 próby logowania. W programie wśród dyrektyw jest również stała MAX_NUMBER_OF_MESSAGES określająca maksymalną możliwą liczbę różnych typów wiadomości (domyślnie 100), czy stała SERVER_STANDBY_TIME, która oznacza maksymalną ilość sekund jaką serwer może jeszcze działać po wylogowaniu się z systemu ostatniego zalogowanego użytkownika (domyślnie jest to 10 sekund).

## 2. Struktura komunikatu

```
struct msgbuf
{
   long type;
   int id;
   char name[256];
   int number;
   int number2;
   char msg[256];
   int messages_types_ids[MAX_NUMBER_OF_MESSAGES];
};
```

- type – typ wiadomości (wspierany przez funkcje systemowe) pozwalający na wysyłkę komunikatów tylko do danych klientów. Wartością typu wysyłanego komunikatu z serwera jest suma identyfikatora klienta, do którego kierowana jest wiadomość, oraz iloczynu stałej określającej dane działanie i maksymalnej liczbie klientów w systemie (MAX_NUMBER_OF_USERS). W przypadku wysyłki żądania od klienta do serwera, typem wiadomości jest odpowiednia unikalna wartość tego działania (są one określone w dyrektywach jako stałe). Taki sposób doboru typów wiadomości gwarantuje ich unikalność.
- id – identyfikator klienta, który wysyła dany komunikat, bądź identyfikator klienta, do którego serwer wysyła komunikat.
- name – identyfikator klienta, który wysyła dany komunikat, bądź identyfikator klienta, do którego serwer wysyła komunikat.
- number – identyfikator nowo utworzonego typu wiadomości / przekazywana liczba różnych typów wiadomości / identyfikator typu wiadomości, do którego klient się rejestruje (subskrybuje) / typ nowo utworzonej wiadomości / identyfikator blokowanego użytkownika
- number2 – liczba pożądanych wiadomości do odbioru w przypadku subskrypcji przejściowej / priorytet nowo utworzonej wiadomości
- msg – rodzaj subskrypcji / wiadomość zwrotna od serwera / nazwa nowo utworzonego typu wiadomości / treść rozgłaszanej wiadomości
- messages_types_ids – lista identyfikatorów różnych typów wiadomości

## 3. Typy komunikatów

Poniżej znajdują się różne rodzaje komunikatów, wraz z wyszczególnionymi elementami struktury msgbuf, które są przez nie wykorzystywane, jak również informację o tym kto jest nadawcą danego komunikatu.

| typ komunikatu | opis | pola struktury | nadawca |
|---|---|---|---|
| LOGIN | Logowanie do systemu | id, name | klient |
| id + ANSWER_LOGIN * MAX_NUMBER_OF_USERS | Informacja zwrotna w sprawie logowania do systemu | id, name | serwer |
| LOGOUT | Wylogowanie z systemu | id, name | klient |
| MESSAGE_TYPE | Tworzenie nowego typu wiadomości | id, name, msg, number | klient |
| id + ANSWER_MESSAGE_TYPE * MAX_NUMBER_OF_USERS | Informacja zwrotna w sprawie tworzenia nowego typu wiadomości | msg | serwer |
| id + NEW_MESSAGE_TYPE * MAX_NUMBER_OF_USERS | Informacja do wszystkich klientów o nowo stworzonym typie wiadomości | id, name, number | serwer |
| DISPLAY_TYPES | Wyświetlenie typów wiadomości | id, name, | klient |
| id + ANSWER_DISPLAY_TYPES * MAX_NUMBER_OF_USERS | Komunikat zwrotny od serwera z listą wszystkich typów wiadomości | number, messages_types_ids | serwer |
| REGISTRATION | Rejestracja do systemu rozgłaszania | id, name, number, number2, msg | klient |
| id + ANSWER_REGISTRATION * MAX_NUMBER_OF_USERS | Informacja zwrotna o rejestracji do danego typu wiadomości | msg | serwer |
| BLOCKING | Blokowanie danego użytkownika | id, name, number | klient |
| id + ANSWER_ BLOCKING * MAX_NUMBER_OF_USERS | Informacja zwrotna o blokowaniu danego innego użytkownika | id, name, number | serwer |
| SEND_MESSAGE | Rozgłoszenie nowej wiadomości | id, name, msg, number, number2 | klient |
| id + SENDING * MAX_NUMBER_OF_USERS | Odbiór rozgłoszonej wiadomości | id, msg, number2 | serwer |

4. **Definicje używanych stałych do określania unikalnych typów komunikatów**

   Stałe, które można dowolnie konfigurować:

   - MAX_NUMBER_OF_USERS = 5 (domyślnie)
   - MAX_LOGIN_ATTEMPTS = 3 (domyślnie)
   - MAX_NUMBER_OF_MESSAGES = 100 (domyślnie)
   - SERVER_STANDBY_TIME = 10 (domyślnie)

   Stałe, których nie należy modyfikować:

   - LOGIN = 1
   - ANSWER_LOGIN = 9
   - MESSAGE_TYPE = 2
   - ANSWER_MESSAGE_TYPE = 10
   - DISPLAY_TYPES = 3
   - ANSWER_DISPLAY_TYPES = 11
   - REGISTRATION = 4
   - ANSWER_REGISTRATION = 12
   - BLOCKING = 5
   - ANSWER_BLOCKING = 13
   - NEW_MESSAGE_TYPE = 14
   - SEND_MESSAGE = 6
   - LOGOUT = 7
   - SENDING = 8

5. **Poprawność danych**

   Identyfikator logującego się klienta do serwisu musi mieć być większy od 0. W przypadku wyboru rodzaju subskrypcji użytkownik musi podać literę „s” bądź „p”, oznaczające odpowiednio subskrypcję stałą lub przejściową. W czasie tworzenia nowego typu wiadomości w systemie użytkownik musi podać literę „s” bądź „a”, oznaczające odpowiednio samodzielną rejestracje nowego typu lub systemową (unikalną), za której poprawność odpowiada serwer. Jeśli użytkownik wybierze pierwszą z tych dwóch opcji, to w kolejnym etapie wprowadzany identyfikator musi być różny od zera, a wprowadzana nazwa inna niż „systemowy typ”, gdyż ta zarezerwowana jest dla nowo tworzonego typu wiadomości przez serwer. Nazwy czy wiadomości to ciągi znaków, których długości nie mogą przekroczyć 255 znaków.

## 6. Scenariusze komunikacji

- **Logowanie**

  Klient:

  type = LOGIN

  id = [tworzony identyfikator użytkownika podczas próby logowania]

  name = [tworzona nazwa użytkownika podczas próby logowania]

  Serwer:

  type = id + ANSWER_LOGIN * MAX_NUMBER_OF_USERS

  id = identyfikator podany przez proces próbujący się zalogować

  name = [informacja zwrotna od serwera] ("Zalogowano pomyślnie" / "W bazie widnieje już użytkownik o takim identyfikatorze" \ "W bazie widnieje już użytkownik o takiej nazwie" \ "W bazie widnieje już taki użytkownik")

- **Wyświetlenie typów wiadomości**

  Klient:

  type = DISPLAY_TYPES

  id = [identyfikator użytkownika, który wysyła żądanie]

  name = [nazwa użytkownika, który wysyła żądanie]

  Serwer:

  type = id + ANSWER_DISPLAY_TYPES * MAX_NUMBER_OF _USERS

  messages_types_ids = lista identyfikatorów dostępnych typów

  number = liczba wszystkich typów wiadomości w systemie

- **Rejestracja do systemu rozgłaszania**

  Klient:

  type = REGISTRATION

  id = [identyfikator użytkownika, który wysyła żądanie]

  name = [nazwa użytkownika, który wysyła żądanie]

  number = [identyfikator typu wiadomości, który użytkownik chce zacząć subskrybować]

  msg = [rodzaj subskrypcji] („s” / „p”)

  number2 = [liczba wiadomości do odbioru w przypadku wyboru subskrypcji przejściowej]

Serwer:

type = id + ANSWER_REGISTRATION * MAX_NUMBER_OF _USERS

msg = [informacja zwrotna od serwera] („Dodano do bazy subskrybentów" / "Użytkownik subskrybuje już ten typ wiadomości" / "W bazie nie ma takiego identyfikatora typu wiadomości")

- **Rejestracja typu wiadomości do systemu rozgłaszania**

Klient:

type = MESSAGE_TYPE

id = [identyfikator użytkownika, który wysyła żądanie]

name = [nazwa użytkownika, który wysyła żądanie]

msg = [nazwa nowo tworzonego typu wiadomości]

number = [identyfikator nowo tworzonego typu wiadomości]

Serwer (informacja zwrotna do klienta tworzącego nowy typ wiadomości):

type = id + ANSWER__MESSAGE_TYPE * MAX_NUMBER_OF _USERS

msg = [informacja zwrotna od serwera] "Istnieje już typ o podanym identyfikatorze." / "Istnieje już typ o podanej nazwie."/ "Podany typ już istnieje." / "Utworzono nowy typ wiadomości")

Serwer (informacja o nowym typie wiadomości do pozostałych użytkowników):

type = id + NEW__MESSAGE_TYPE * MAX_NUMBER_OF _USERS

id = [identyfikator użytkownika, który stworzył nowy typ wiadomości]

number = [identyfikator nowo utworzonego typu wiadomości]

name = [nazwa nowo utworzonego typu wiadomości]

- **Rozgłoszenie nowej wiadomości**

Klient:

type = SEND_MESSAGE

id = [identyfikator użytkownika, który wysyła żądanie]

name = [nazwa użytkownika, który wysyła żądanie]

msg = [treść rozgłaszanej wiadomości]

number = [typ rozgłaszanej wiadomości]

number2 = [priorytet rozgłaszanej wiadomości]

Serwer (wysyłka wiadomości do innych klientów):

type = id + SENDING * MAX_NUMBER_OF _USERS

id = [identyfikator użytkownika, który rozgłasza wiadomość]

msg = [treść rozgłaszanej wiadomości]

number2 = [priorytet rozgłaszanej wiadomości]

- **Blokowanie innego użytkownika**

  Klient:

  type = BLOCKING

  id = [identyfikator użytkownika, który wysyła żądanie]

  name = [nazwa użytkownika, który wysyła żądanie]

  number = [identyfikator blokowanego użytkownika]

  Serwer (wysyłka wiadomości do innych klientów):

  type = id + ANSWER_BLOCKING * MAX_NUMBER_OF _USERS

  id = [identyfikator użytkownika, który blokuje]

  name = [nazwa użytkownika, który blokuje]

  number = [zmienna określające czy użytkownik o podanym identyfikatorze istnieje] (1 / 0)

- **Wylogowanie z systemu**

  Klient:

  type = LOGOUT

  id = [identyfikator użytkownika, który wysyła żądanie]

  name = [nazwa użytkownika, który wysyła żądanie]