

# PSiW - Protokół

Anna Ogorzałek 151805

Bartosz Nowak 151891

## 1 Komunikacja między serwerem i klientem

### 1.1 Wstęp

Cała komunikacja opiera się wyłącznie na jednej kolejce komunikatów IPC. Klucz jest stałą dostępną dla wszystkich programów. Serwer oczekuje na jakiekolwiek wiadomości, natomiast wysyłając komunikaty do użytkownika, przypisuje do typu konkretne ID użytkownika - czyli PID procesu. Użytkownik odbiera wiadomości oczekując na wiadomość z jego ID.

### 1.2 Polecenia

Przy włączeniu programu, jedyne co może zrobić użytkownik to wprowadzenie nazwy użytkownika i hasła w celu zalogowania się. Komendy z których mogą korzystać użytkownicy, dostępne są dopiero po zalogowaniu się na konto. W celu ich wyświetlenia można użyć komendy: "!help" Wykonanie każdego polecenia wymaga rozróżnienia tych poleceń przez serwer, dlatego każde żądanie ze strony użytkownika jest wysyłane z konkretnym typem, dla każdego z poleceń.

Komendy razem z funkcjonalnością i typem:

polecenie	funkcja	typ
—	loguje użytkownika	LOGIN_TYPE
!help	wyświetla dostępne komendy	brak
!ulist	wyświetla listę zalogowanych użytkowników	USERS_LIST_TYPE
!glist	wyświetla dostępne grupy	GROUP_LIST_TYPE
!guser	wyświetla użytkowników w danej grupie	GROUP_LIST_TYPE
!gjoin	pozwala na dołączenie do danej grupy	GROUP_JOIN_TYPE
!gexit	pozwala na opuszczenie danej grupy	GROUP_EXIT_TYPE
!dm	wysyła wiadomość do danego użytkownika	DIRECT_MESSAGE_TYPE
!gm	wysyła wiadomość do grupy	GROUP_MESSAGE_TYPE
!muteuser	blokuje konkretnego użytkownika	MUTE_USER_TYPE
!unmuteuser	odblokowuje użytkownika	UNMUTE_USER_TYPE
!mutegroup	blokuje konkretną grupę	MUTE_GROUP_TYPE
!unmutegroup	odblokowuje grupę	UNMUTE_GROUP_TYPE
!logout	wylogowuje użytkownika i wyłącza program	LOGOUT_TYPE

## 1.3 Komunikacja

Struktura przekazywanych komunikatów jest następująca:

```
1 struct msgbuf{
2     long type;
3     int error;
4     int receiverId;
5     int senderId;
6     char message[1024];
7 };
```

za jej pomocą przekazywane są wszystkie informacje pomiędzy użytkownikiem a serwerem.

### 1.3.1 Logowanie

Użytkownik wprowadza nazwę użytkownika i hasło, które zostaje przesłane w formie username;password. Serwer po otrzymaniu wiadomości sprawdza czy istnieje taki użytkownik (na podstawie danych odczytanych wcześniej z pliku), następnie sprawdza czy hasło jest poprawne, a także czy użytkownik nie jest już zalogowany, i czy nie przekroczył dostępnych prób logowania. Serwer odsyła zaistniały błąd lub potwierdzenie zalogowania.

**Wymagane:** long type, int senderId (PID), int receiverId (child PID), char message[1024]

### 1.3.2 Lista użytkowników

Do serwera przesyłana zostaje informacja z ID użytkownika i typem zapytania. Serwer po otrzymaniu wiadomości, przeszukuje wszystkich zalogowanych użytkowników i odsyła informację w formie user1;user2;...;usern, co wyświetlane jest u użytkownika odpowiednio sformatowane.

**Wymagane:** long type, int senderId (PID)

### 1.3.3 Dostępne grupy

Do serwera przesyłana zostaje informacja z ID użytkownika i typem zapytania. Serwer po otrzymaniu wiadomości, przeszukuje wszystkie istniejące grupy i odsyła informację w formie group1;group2;...;groupn co wyświetlane jest u użytkownika odpowiednio sformatowane.

**Wymagane:** long type, int senderId (PID)

### 1.3.4 Użytkownicy danej grupy

Do serwera przesyłana zostaje informacja z ID użytkownika, typem zapytania i nazwą grupy. Serwer po otrzymaniu wiadomości, wyszukuje podaną grupę - odsyłając błąd jeśli nie istnieje. Następnie sprawdza czy w grupie są użytkownicy i jeśli wszystko jest w porządku, odsyła informację w formie user1;user2;...;usern, co wyświetlane jest u użytkownika odpowiednio sformatowane.

**Wymagane:** long type, int senderId (PID), char message[1024]

### 1.3.5 Dołączanie do grupy

Do serwera przesyłana zostaje informacja z ID użytkownika, typem zapytania i nazwą grupy. Serwer po otrzymaniu wiadomości, wyszukuje podaną grupę - odsyłając błąd jeśli nie istnieje. Następnie sprawdza czy w grupie nie ma już tego użytkownika, a potem czy jest jakieś wolne miejsce, do którego może dołączyć użytkownika. Serwer odsyła zaistniały błąd lub potwierdzenie dołączenia do grupy.

**Wymagane:** long type, int senderId (PID), char message[1024]

### **1.3.6 Opuszczanie grupy**

Do serwera przesyłana zostaje informacja z ID użytkownika, typem zapytania i nazwą grupy. Serwer po otrzymaniu wiadomości, wyszukuje podaną grupę - odsyłając błąd jeśli nie istnieje. Następnie sprawdza czy użytkownik znajduje się w tej grupie, po czym go z niej usuwa. Serwer odsyła zaistniały błąd lub potwierdzenie usunięcia użytkownika z grupy.

**Wymagane:** long type, int senderId (PID), char message[1024]

### **1.3.7 Wysyłanie wiadomości**

**Wymagane:**

### **1.3.8 Wysyłanie wiadomości do grupy**

**Wymagane:**

### **1.3.9 Blokowanie użytkownika**

**Wymagane:**

### **1.3.10 Odblokowanie użytkownika**

**Wymagane:**

### **1.3.11 Blokowanie grupy**

**Wymagane:**

### **1.3.12 Odblokowanie grupy**

**Wymagane:**

### **1.3.13 Wylogowanie użytkownika**

Do serwera przesyłana zostaje informacja z ID użytkownika, typem zapytania i nazwą grupy. Serwer usuwa użytkownika z aktywnych użytkowników i wysyła potwierdzenie wykonania żądania.

**Wymagane:** long type, int senderId (PID)