Projekt zaliczeniowy
Systemy Operacyjne 2

Arkadiusz Danilecki, Mateusz Hołenko, Anna Kobusińska, Andrzej Stroiński, and Piotr Zierhoffer

Instytut Informatyki, Politechnika Poznańska

{adanilecki,mholenko,akobusinska,astroinski,pzierhoffer}
@cs.put.poznan.pl

1 Wstęp

Celem projektu jest stworzenie programu pozwalającego komunikację pomiędzy użytkownikami (czatu) opartego o architekturę rozproszonych serwerów. W ramach realizacji projektu konieczne będzie opracowanie protokołu komunikacyjnego, stworzenie aplikacji klienta i serwera, oraz przygotowanie prostej dokumentacji.

2 Opis środowiska

System czatu opierać się ma na wielu klientach łączących się z siecią serwerów. Każdy klient ma wspierać wysyłanie dwóch typów wiadomości:

- publicznych — widocznych dla wszystkich użytkowników czatu
- prywatnych — skierowanych do konkretnego użytkownika i widocznych tylko przez niego (użytkownicy rozróżniani powinni być po unikalnych nazwach).

Każdy serwer podzielony powinien być na niezależne kanały (pokoje) — wiadomości publiczne wysyłane w ramach kanału powinny być widoczne dla innych uczestników tego kanału, jednak nie powinny dotrzeć do pozostałych użytkowników podłączonych do serwera. Kanały nie mają ograniczać się do jednego serwera, tzn. jeżeli istnieje kanał o tej samej nazwie na dwóch serwerach to stanowi on jeden kanał — wiadomości wysyłane do pierwszego serwera muszą zostać przekazane również do drugiego.

Kanał ma być tworzony dynamicznie, w momencie, kiedy podłącza się do niego pierwszy klient. Kanały mają być jednoznacznie identyfikowane przez nazwę.

Każdy klient podłączony jest z jednym serwerem, jednak do serwera podłączonych jest dowolnie wiele klientów. Każdy serwer, na podstawie informacji z centralnego repozytorium, komunikuje się z wszystkimi innymi dostępnymi serwerami. W ten sposób klienci komunikują się z klientami podłączonymi do dowolnego serwera.

3 Wymagania ogólne

1. Wszystkie komponenty systemu napisane są w języku C.
2. Komunikacja między aplikacjami odbywa się **wyłącznie** za pomocą wyspecyfikowanych mechanizmów.
3. Praca jest w stu procentach **samodzielna**.
4. Komponenty muszą być kompatybilne w ramach grupy laboratoryjnej, każdy serwer i klient musi współpracować ze wszystkimi innymi serwerami i klientami.

4 Komponenty systemu

4.1 Klient

Aplikacja kliencka służy jako interface dla użytkownika. Interface zrealizowany może być w dowolnej technologii, musi być czytelny i wygodny w obsłudze.

Funkcje realizowane przez klienta:

- przyjęcie nazwy użytkownika,
- zarejestrowanie użytkownika w zadanym serwerze,
- wyświetlanie listy użytkowników,
- wyświetlanie czatu (każda wiadomość to czas wysłania i nazwa użytkownika),
- możliwość wysyłania wiadomości czatu,
- możliwość odebrania wiadomości prywatnej (wyraźnie rozróżnionej),
- możliwość wysłania wiadomości prywatnej do dowolnego użytkownika,
- możliwość wylogowania z systemu,
- możliwość wyboru kanału czatu,
- odpowiadanie na komunikaty *heartbeat* serwera.

Klient porozumiewa się wyłącznie z serwerem, z którym się połączył. Całość komunikacji odbywa się za pomocą kolejek komunikatów. Odświeżanie interface'u odbywa się automatycznie.

Adres serwera (numer kolejki komunikatów), z którym klient ma się połączyć, podawany jest przez użytkownika w trakcie uruchamiania lub działania aplikacji (nie może być wkompilowany w wersję binarną aplikacji).

4.2 Repozytorium

Centralne repozytorium danych to obszar pamięci (lub więcej obszarów) współdzielony przez wszystkie serwery. Jego identyfikator jest przekazywany serwerowi w momencie uruchomienia przez administratora serwera (użytkownika, który go uruchamia).

Przechowuje następujące dane:

- lista identyfikatorów kolejek komunikatów, na których nasłuchują serwery,
- lista zalogowanych użytkowników, wraz z ich serwerem macierzystym,
- informacja o aktywnych kanałach w ramach poszczególnych serwerów,
- ewentualna dodatkowa konfiguracja.

Dostęp do segmentów pamięci współdzielonej musi być wykonywany w sposób zabezpieczony semaforami. Każdy obszar pamięci traktowany jest jako sekcja krytyczna z możliwością dostępu jednego użytkownika.

Obszar pamięci współdzielonej **nie posiada** aplikacji, która zarząda dostępem do niego — każdy serwer odpowiedzialny jest za odpowiednie oprogramowanie dostępu do współdzielonego obszaru z uwzględnieniem zapewnienia wykluczania.

4.3 Serwer

Serwer realizuje większość logiki systemu. Odpowiada za komunikację z użytkownikiem, repozytorium oraz innymi serwerami.

Każdy serwer może obsłużyć do 20 klientów. Wszystkie serwery nawiązują łączność z wszystkimi innymi serwerami, na podstawie najbardziej aktualnej listy.

Serwer winien otrzymać unikalny numer kolejki komunikatów oraz identyfikator segmentu pamięci współdzielonej. Podanie zajętego identyfikatora kolejki powinno skutkować zamknięciem serwera.

Czynności wykonywane przez serwer:

- wpisanie się do rejestru,
- usunięcie się z rejestru (po wpisaniu komendy i/lub złapaniu odpowiedniego sygnału)
- odbiór wiadomości czatu i wysyłanie ich do wszystkich innych serwerów,
- przesłanie odebranych wiadomości czatu do wszystkich swoich użytkowników,
- odbiór wiadomości prywatnych i wysyłanie ich do serwerów odpowiednich klientów (bez rozgłaszania, na podstawie aktualnych wiadomości),
- rejestrowanie i wyrejestrowanie użytkowników,
- weryfikacja obecności klientów (mechanizm *heartbeat*),
- w wypadku kilkukrotnego niepowodzenia w skontaktowaniu się z innym serwerem, usunięcie jego wpisu z rejestru.

Klienci niewykryci przez mechanizm *heartbeat* nie mogą komunikować się z serwerem bez ponownego zalogowania — serwer powinien odrzucać ich komunikaty, informując o błędzie (nawet, jeżeli de facto nie przerwali działania).

Mechanizm logowania Każdy serwer winien logować do pliku /tmp/czat.log informacje o następujacych zdarzeniach:

- zarejestrowanie serwera
- wyrejestrowanie serwera
- zalogowanie klienta
- wylogowanie klienta
- usunięcie klienta przez *heartbeat*
- usunięcie nieodpowiadającego serwera

Dostęp do pliku logu musi być synchronizowany za pomocą semaforów.

5 Oddanie projektu

Projekt musi zostać dostarczony w terminie określonym przez prowadzącego zajęcia drogą e-mailową. Winien on mieć postać archiwum o nazwie postaci `nazwisko_imie_indeks.tar.gz`.

Archiwum projektu musi zawierać następujące elementy:

- plik README zawierający informację o autorze (imię, nazwisko, e-mail, nr indeksu), opis wszystkich plików wchodzących w skład projektu oraz instrukcję skompilowania, uruchomienia oraz użytkowania komponentów,
- plik Makefile,
- opcjonalne skrypty uruchomieniowe,
- kody źródłowe wszystkich aplikacji,
- wszystkie dodatkowe pliki wymagane do działania aplikacji.

Pliki źródłowe muszą zawierać komentarze dla każdej funkcji (wraz z opisem parametrów), struktury (z opisem poszczególnych pól) i dla bardziej skomplikowanych fragmentów kodu.

Archiwum tar.gz **nie może** zawierać plików wykonywalnych!

Niedostarczenie projektu w terminie powodować będzie obniżeniem oceny, zgodnie z zasadami podanymi przez prowadzącego.

6 Ułatwienia projektu

Kosztem obniżenia maksymalnej oceny z projektu możliwa jest rezygnacja z części funkcjonalności.

6.1 Projekt na 4

- brak wykrywania awarii serwera
- brak obsługi pliku logu

6.2 Projekt na 3

- klient synchroniczny, odświeżający się tylko na żądanie klienta
- brak komunikacji z innymi projektami z grupy laboratoryjnej (niezgodność z protokołem)