TIN – Projekt wstępny

Program obsługujący prosty kanał P2P
w oparciu o protokół BitTorrent

Semestr 21L

Dąbek Marcelina | Okrutny Aleksandra | Santorowska Zuzanna | Such Jan

Opis funkcjonalny

Program zostanie napisany przy wykorzystaniu protokołu IPv6 oraz TCP, będzie on obsługiwany przy użyciu terminala tekstowego.

Aplikacja będzie umożliwiała:

- udostępnienie przez użytkownika zasobu (oryginału)
- polecenie pobrania nazwanego zasobu ze zdalnego węzła
- pobranie zasobu ze zdalnego węzła (utworzenie lokalnej kopii zasobu)
- rozsyłanie informacji o lokalnie posiadanych zasobach.

Użytkownik będzie miał możliwość unieważnić swój zasób (w przypadku, gdy posiada oryginał zasobu). W przypadku unieważnienia przez użytkownika oryginalnego zasobu system losowo wybierze użytkownika, którego kopia zasobu od tej pory będzie uznawana za oryginał.

Rozgłaszania posiadania zasobu (oryginału i kopii) oraz unieważnienie zasoby będzie rozgłaszane odpowiednim komunikatem.

Zasób jest identyfikowany poprzez nazwę, dlatego próba dodania do sieci zasobu o już istniejącej nazwie będzie sygnalizowana błędem.

Stosowane protokoły komunikacyjne

Nasz projekt realizowany będzie w oparciu o protokół BitTorrent.

Protokół BitTorrent to protokół umożliwiający przesyłanie plików w sieci peer-to-peer, służy do decentralizacji dostępu do zasobu w sieci. Gdy jeden z jej użytkowników udostępnia jakiś zasób i wiele osób chce ten zasób otrzymać wtedy można rozdystrybuować zasoby do innych maszyn, aby dało się go ściągać również z innych miejsc niż tylko z oryginalnej maszyny. Plik jest dzielony na mniejsze kawałki i przechowywany w roju, czyli grupie maszyn. Można taki plik ściągać symultanicznie.

Meta-informacje dotyczące pobieranego zasobu przechowywane są w pliku .torrent. Jest on kluczem do pobrania właściwej treści zasobu. Użytkownik ściąga plik .torrent z serwera indeksowego, a następnie otwiera go w kliencie BitTorrent który pozwala na automatyzację reszty procesu - łączy się z odpowiednimi Trackerami aby ściągnąć poszczególne fragmenty zasobu.

Tracker to serwer asystujący przy komunikacji peer-to-peer. Posiada informację na temat lokalizacji poszczególnych fragmentów pliku i ich kopii u poszczególnych użytkowników oraz tego, który z użytkowników jest aktualnie aktywny. Udostępnia klientowi adresy IP maszyn, na których znajdują się fragmenty pliku, który chce pobrać. W zamian za to klient co jakiś czas wysyła Tracker’owi informację na temat postępów w pobieraniu. Dzięki tej komunikacji, tracker może w trakcie pobierania udostępnić adresy IP nowych maszyn, które posiadają kopię fragmentu pliku, a pojawiły się po tym jak pobieranie się rozpoczęło.

Użycie protokołu BitTorrent w przypadku naszego projektu, bardzo pozytywnie wpływa na jego wydajność. Zastosowanie wielu maszyn, tzw. seed'ów, udostępniających pliki do pobrania, zamiast jednego głównego serwera, umożliwia symultaniczne pobieranie

zasobów, co znacznie skraca trwanie tych operacji. Zabieg ten równocześnie chroni przed tzw. “slashdot effect” gdy wiele osób na raz próbuje pobrać dane z tego samego miejsca oraz zapewnia większą redundancję danych w systemie.

Zastosowanie Trackera umożliwia ciągłą kontrolę nad liczbą kopii plików w systemie i ich lokalizacją, a dzięki komunkacji z peer’ami pobierającymi zasoby może kierować przepływem danych w taki sposób by zasoby były pobierane jak najbardziej równomiernie ze wszystkich oferujących je maszyn, co również zapewnia szybszy transfer.

W projekcie użyjemy również protokołów TCP oraz IPv6.

Protokół TCP to strumieniowy protokół warstwy transportowej, działający w trybie klient-serwer. TCP w przeciwieństwie do UDP gwarantuje wyższym warstwom komunikacyjnym dostarczenie wszystkich pakietów w całości, z zachowaniem kolejności i bez duplikatów zapewniając w ten sposób wiarygodność połączenia. Jest to kluczowe w przypadku sieci opierającej się na połączeniach z wieloma różnymi użytkownikami na raz i odbieraniu lub wysyłaniu fragmentów plików od/do nich.

IPv6 to najnowsza wersja protokołu IP. Jest to protokół warstwy sieciowej zapewniający system identyfikacji i lokalizacji komputerów w sieci oraz kierujący ruchem w Internecie. Protokół ten umożliwia m.in Multicasting, czyli transmisję pakietu do wielu miejsc w jednej operacji wysłania.

Podział na moduły i komunikację między nimi

Do realizacji współbieżności wykorzystana zostanie wielowątkowość. Każdy z modułów programu będzie działał w oddzielnym wątku w celu zapewnienia asynchroniczności wykonywania operacji, natomiast komunikacja pomiędzy modułami / wątkami będzie się odbywać poprzez prostą kolejkę wiadomości.

W skład modułów wchodzić będą:

- Interfejs
  - ***Obsługa wejścia*** – nawigacja w menu, wpisywanie nazw plików etc.
  - ***Wypisywanie na ekran*** – wyświetlanie menu, pasków postępu, komunikatów

- Usługa P2P
  - ***Komunikacja z trackerem*** – zbieranie informacji o dostępnych plikach

  - ***Ustanawianie połączeń z peer'ami*** – przy pomocy socketów oraz funkcji select(), komunikacja z innymi peer’ami, zarządzanie pobieraniem

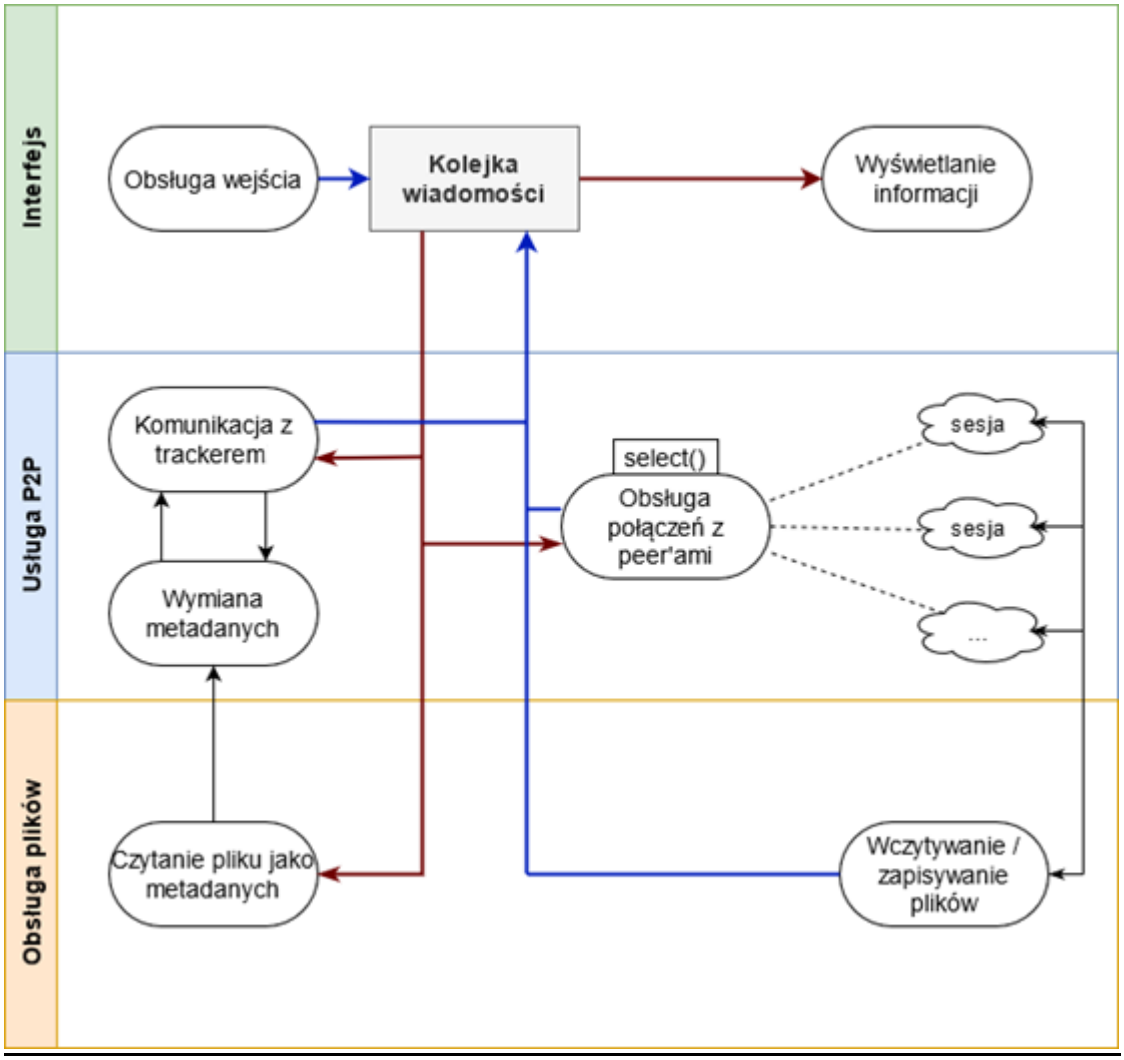
- Wymiana metadanych
  - ***Czytanie pliku jako metadanych*** – tworzenie plików .torrent
  - ***Wczytywanie/zapisywanie do pliku*** – przy pomocy zwykłego fstreama, współpracujący z *ustanawianiem połączeń*

Jako główny sposób komunikacji międzymodułowej wykorzystana zostanie wcześniej wspomniana **kolejka wiadomości**. Celem tego rozwiązania jest utworzenie kanału dla

komunikatów, których przetworzenie nie będzie priorytetem. Będzie to prosta struktura (np. tablica), która będzie przechowywać wiadomości w postaci ciągów znaków.

Każda wiadomość będzie posiadała informacje o odbiorcy (ID modułu), typie oraz treści. Moduły będą okresowo skanować ostatnie puste pole w celu sprawdzenia, czy nadeszła dla nich wiadomość (innymi słowy polling, ale wykonywany sporadycznie by nie zmniejszać wydajności aplikacji).

Diagram modułów



Zarys koncepcji implementacji

Zadanie zostanie zaimplementowane przy użyciu języka C z wykorzystaniem standardowego API gniazd BSD/Unix. Wielowątkowość zostanie obsłużona przy użyciu biblioteki pthread.h. Testy zostaną napisane we frameworku Unity. Do realizacji interfejsu użytkownika użyjemy terminala w trybie „raw” (sterowanie podobne do np. vi).

Projekt będzie zarządzany z wykorzystaniem gita i GitHuba, a sam kod zostanie napisany przy wsparciu narzędzia Clion.

Spis treści

Opis funkcjonalny 1
Stosowane protokoły komunikacyjne 1
Podział na moduły i komunikację między nimi.....2
Diagram modułów 3
Zarys koncepcji implementacji 4