|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr ćwicz.  4-7 | Data  30.11.2016 | Szymon Królikowski **127055** Daniel Pawlikowski **126899** | Wydział Elektryczny | Semestr  III | grupa I4-2 |
| **Implementacja protokołu warstwy zastosowań – sprawozdanie końcowe.** | | | | | |

# Implementowany protokół: FTP

Oprogramowanie klienta zostało zaimplementowane przez Szymona Królikowskiego

Oprogramowanie serwera zostało zaimplementowane przez Daniela Pawlikowskiego

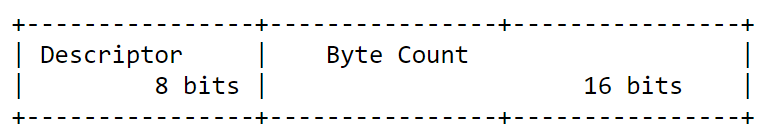
# Specyfikacja protokołu

**FTP (File Transfer Protocol)** - protokół komunikacyjny typu klient-serwer wykorzystujący protokół TCP według modelu TCP/IP, umożliwiający dwukierunkowy transfer plików w układzie serwer FTP–klient FTP. FTP jest protokołem 8-bitowym. Do komunikacji wykorzystywane są dwa połączenia TCP. Jedno z nich jest połączeniem kontrolnym za pomocą którego przesyłane są polecenia, a drugie służy do transmisji danych.

Połączenie za pomocą protokołu FTP może działać w dwóch trybach: aktywnym i pasywnym:

* 1. Aktywne – port 21 do przesyłu poleceń zestawiony przez serwer, oraz port 20 do przesyłu danych zestawiony przez klienta.
  2. Pasywne – port 21 do przesyłu poleceń oraz port większy niż 1024 do przesyłu danych, obydwa zestawione przez klienta.

Budowa nagłówka FTP:



Descriptor – zawiera jeden z 4 kodów:

128 End of data block is EOR

64 End of data block is EOF

32 Suspected errors in data block

16 Data block is a restart marker

Byte Count – zawiera liczbe danych zawartą w bloku danych

W FTP wyróżniamy takie rodzaje komunikatów jak:

-ACCESS CONTROL COMMANDS np.:

1. USER

2. PASS

3. ACCT

-TRANSFER PARAMETER COMMANDS np.:

1. PORT

2. PASV

3. STRU

-FTP SERVICE COMMANDS np.:

1.RETR

2.STOR

3.STOU

# Specyfikacja implementacji

Klient:

Elementami podlegającymi implementacji są:

- klasa Socket, przyjmuje dwa argumenty w konstruktorze: hostIP oraz numer portu

na jakim nasłuchuje dany serwer

- klasa BufferedReader, przyjmuje jako argument nowy obiekt typu

InputStreamReader z podanym w konstruktorze wejściem systemowym w postaci

System.in. Czyta on wprowadzone znaki z konsoli i wczytuje do zmiennej.

Server:

Elementami podlegającymi implementacji są:

- klasa ServerSocket, przyjmuje jeden argument w konstruktorze w postaci portu na

którym nasłuchuje, nawiązywanie połączenia następuje przez metodę accept()

wywołaną z obiektu ServerSocket

Wykorzystany jest stos protokołów to TCP/IP

Struktura komunikatów:

W celach komunikacyjnych zaimplementowane zostały obiekty klasy DataInputStream służący

za wejście obiektu Socket oraz DataOutputStream będący wyjściem tego obiektu. Komunikaty

są przesyłane za pomocą funkcji writeUTF przyjmującej argument String a odbierane po przy

pomocy funkcji readUTF.

Opis mechanizmu transmisji danych:

Do przesyłania danych tworzone jest specjalne połączenie na porcie 1200. Posłużono się tutaj

klasami BufferedOutputStream oraz BufferedInputStream w celu szybkiej serializacji oraz

przesyłu danych. W przypadku przesyłu danych klasa BufferedOutputStream w swoim konstruktorze przyjmuje obiekt OutputStream pochodzacy od socketu klienta natomiast BufferedInputStream przyjmuje w konstruktorze obiekt typu File będący wysyłanym plikiem. W przypadku odbioru i zapisu danych klasa BufferedOutputStream przymuje jako argument ścieżkę zapisu pliku natomiast BufferedInputStream obiekt InputStream pochodzacy od socketu klienta.

Środowiskiem programistycznym został IntelliJ IDEA firmy JetBrains

Język programowania: Java

Biblioteki: Zestaw standardowych bibliotek dostarczonych przez firme Oracle

# Testy

Do przeprowadzonych testów zalicza się:

- test aplikacji w loopback

- test z wykorzystaniem aplikacji RawCap

- analiza pliku utworzonego przez RawCap przy użyciu WireShark

# Podsumowanie

Wnioski:

Protokół FTP jest nieodzownym elementem sieci web, jego implementacja zdecydowanie przyczyniła się do lepszego zrozumienia zagadnienia oraz tego jak działa sam FTP

Do zalet samego FTP można zaliczyć prostotę w urzyciu i funkcjonalność.

Największą wadą protokołu jest brak szyfrowania co powoduje, że dane przesyłane miedzy serwerem a klientem mogą być łatwo wykradzione.

# Bibliografia

RFC 959

Wikipedia.org

stackoverflow.com