



Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №9
з дисципліни «Технології розроблення програмного
забезпечення»

на тему «різні види взаємодії додатків:
client-server, peer-to-peer, service-oriented architecture»
Тема для лабораторного циклу «HTTP-сервер»

Виконала

студентка групи ІА-04

Глушко

Юлія Петрівна

Перевірив

Колеснік Валерій

Миколайович

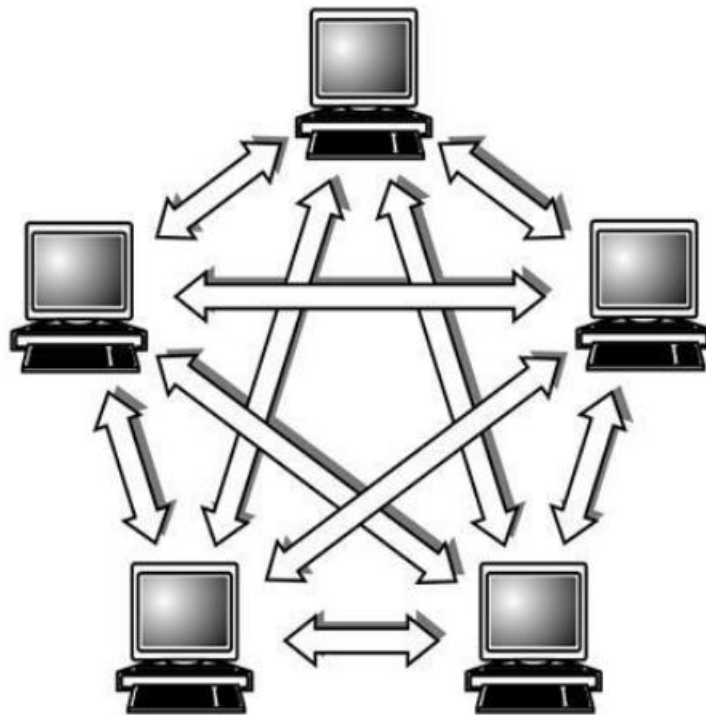
Защищено з балом _____

Київ 2022

Хід роботи :

Додатки типу «peer-to-peer»

Теоретичні відомості



Модель взаємодії додатків типу реєр - to - реєр має на увазі рівноправ'я клієнтських програм і відсутність серверної програми. Усі клієнтські програми контактують між собою для виконання спільних цілей. У таких мережах найчастіше виникають наступні проблеми: синхронізація даних, пошук клієнтських застосувань.

Для пошуку клієнтських застосувань може використовуватися одна загальна адреса, що містить набір адрес підключених клієнтів. Це в деякому роді нагадує сервер. Проте сервер є лише центром сходження безлічі клієнтів, а адреси можуть зберігатися в самих клієнтських застосуваннях. Це називається структуровані однорангові мережі.

Для вирішення проблеми синхронізації даних використовуються традиційні підходи: можливий варіант договору про набір для синхронізації (negotiation), або визначення по деякому алгоритму порівняння даних (hash-алгоритми і інші). Як правило, peer - to - peer застосування є звичайними застосуваннями з продуманою інфраструктурою обміну повідомленнями між клієнтами. Для таких випадків як правило розробляються спеціальні формати і протоколи обміну для структуризації діалогу між клієнтами.

Застосування при реалізації лабораторного циклу

При виконання лабораторного циклу, HTTP-сервер було реалізовано за p2p архітектурою. Лабораторна робота складається із 3 модулів: Клієнтів, Серверу (без якого все також чудово працює. Сервер потрібен лише для того, щоб peer`ам було легше знайти активні підключення) та інформації про мережу (зв'язок між клієнтами (один із одним) та клієнтами із сервером) . Користувачів можна створити декілька, Сервер для їхнього зв'язку – лише один раз.

Кожен peer одночасно є і сервером і клієнтом. При створенні його призначається випадковий порт, який він має прослуховувати як сервер. Якщо на цей порт приходить запит, то він обробляється і генерується відповідь. Запити та відповіді по HTTP/1.1 протоколу.

Сервер працює за такою схемою :

Через графічний інтерфейс передаються потрібні дані (Рис.1), а саме , метод, який потрібно використати (get або post)(регістр не має різниці), url сторінки та порт, на який потрібно зробити запит . Дані валідуються(рис.2,3). Наступні поля – це параметри/значення, які потрібно передати . Також можна побачити номер порта даного peer та кнопку для відправки даних.

Method: Url:

Receiver:

parameter 1: value 1:

parameter 2: value 1:

parameter 3: value 1:

Port 9189

Connection is ready

Рис.1

Method: Url:

Incorrect method

Receiver:

parameter 1: value 1:

parameter 2: value 1:

parameter 3: value 1:

Port 9190

Connection is ready

Рис.2

Method: Url:

Receiver:

NaN!!!

parameter 1: value 1:

parameter 2: value 1:

parameter 3: value 1:

Port 9190

Connection is ready

Рис.3

The screenshot shows a Java IDE with a REST client configuration window and an HTTP response window. The REST client window has the following fields:

- Method:
- Uri:
- Receiver:
- parameter 1:
- parameter 2:
- parameter 3:
- value 1:
- value 1:
- value 1:

The HTTP response window shows the following content:

```
HTTP/1.1 200 OK
Content-Type: html
Content-Length: 228

<DOCTYPE html><html lang="en"><head> <meta charset="UTF-8"> <title>Title</title></head><body><center><h3>Page has been visited 141 ti
mes</h3><br><br><Name> Yulia<br><br>01234<br><br>Message: Hello</center></body></html>
```

The IDE's file explorer shows the project structure:

- src
 - main
 - java
 - org.example.server
 - Server
 - ServerConnectionListener
 - resources
 - test
 - target

The Run console shows the command:

```
D:\Program Files\Java\jdk-17.0.1\bin\java...
```

Рис.4



Рис.5

The screenshot shows a web application window with a light gray background. At the top, there is a standard window title bar with a small icon on the left and three control buttons (minimize, maximize, close) on the right. The main content area contains several input fields and a button. On the left side, there are four labels with corresponding input boxes: 'Method' with 'get' entered, 'Receiver' with '9008' entered, 'parameter 1', 'parameter 2', and 'parameter 3' (all empty). On the right side, there is a 'Url' label with '1/test1.jsp' entered, followed by three 'value 1' labels with empty input boxes. Below these inputs, there is a button labeled 'Ask page'. Underneath the button, the text 'Servers problem!' is displayed in red, with a red horizontal line underneath it. At the bottom center of the form area, the text 'Port 9258' is visible.

Рис.6

Сервер для пошуку активних підключень працює так: потрібно зробити запит на сервер (будь-який метод, будь-який url, замість порту вказати “server”(регістр не має значення)). Після цього дане підключення буде додано до списку активних, та надіслано йому список інших активних підключень . Якщо реєр стає не активним, на сервер автоматично направляється запит, що сповіщає серверу, що дане підключення потрібно видалити із списку активних.

Рисунок 7 – підключення із Google Chrome до активного peer

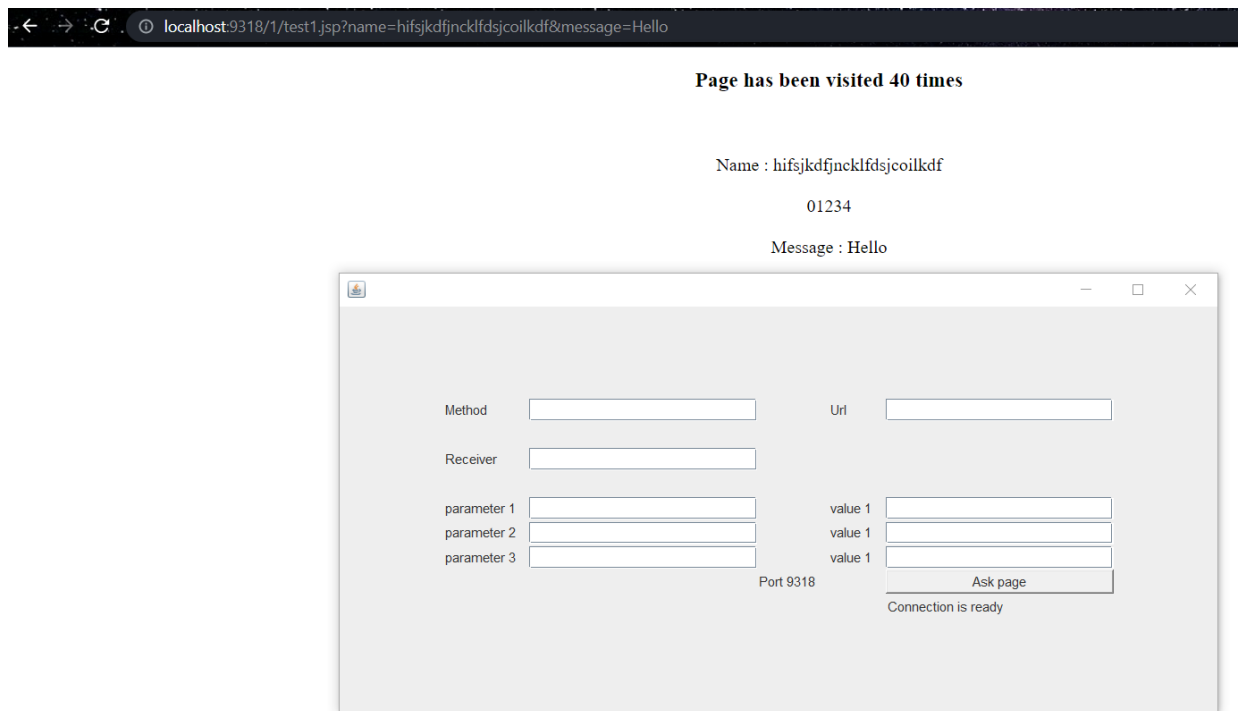


Рис.7

Висновок : під час виконання даної лабоаторної роботи було досліджено різні види взаємодії додатків, а саме client-server, peer-to-peer, service-oriented architecture. Тема лабораторного практикуму була реалізована , як peer-to-peer застосунок