



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  
ТРПЗ  
**Тема:** Взаємодія компонентів системи. HTTP-сервер

Виконав:  
студент групи ІА-31  
Машин Д. І.

Київ 2025

**Мета:** Вивчити види взаємодії додатків (Client-Server, Peer-to-Peer, Service-oriented Architecture), та реалізувати в проєктованій системі одну із архітектур.

### **Завдання**

- Ознайомитись з короткими теоретичними відомостями.
- Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
- Реалізувати функціонал для роботи в розподіленому оточенні відповідно до обраної теми.
- Реалізувати взаємодію розподілених частин:
  - *Для клієнт-серверних варіантів:* реалізація клієнтської і серверної частини додатків, а також загальної частини (middleware); зв'язок клієнтської і серверної частин за допомогою WCF, TcpClient, .NET-Remoting на розсуд виконавця.
  - *Для однорангових мереж:* реалізація взаємодії клієнтських додатків за допомогою WCF Peer to peer channel.
  - *Для SOA додатків:* реалізація сервісу, що надає послуги клієнтським застосуванням; викладання сервісу в хмару або підняття у вигляді Web Service на локальній машині; використання токенів для передачі даних про автентифікації, двостороннє шифрування.
- Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму класів, яка представляє спроектовану архітектуру. Навести фрагменти програмного коду, які є суттєвими для відображення реалізованої архітектури.

## Реалізації Peer-to-peer

```
@Override
public HttpServletResponse handle(HttpServletRequest request) {
    HttpServletResponse availableServer = requestAvailableServer();
    String[] parts = availableServer.getBody().split( regex: ":" );
    String host = parts[0];
    int port = Integer.parseInt(parts[1]);

    if(Objects.equals(availableServer.getStatus(), b: "404 Not Found")){
        logger.warning( msg: "No server available to redirect request.");
        return null;
    }
    logger.info( msg: "Redirecting request ...");

    try{
        long requestTime = System.currentTimeMillis();
        Socket socket = new Socket(host,port);
        DataOutputStream out = new DataOutputStream(socket.getOutputStream());
        DataOutputStream dataOutputStream = new DataOutputStream(out);
        dataOutputStream.writeUTF(request.toString());
        dataOutputStream.flush();
        InputStream in = socket.getInputStream();
        HttpServletResponse response = parser.parse(in);
        long responseTime = System.currentTimeMillis();
        long time = responseTime - requestTime;
        Statistics.saveExternalStats(availableServer.getBody(),request,response.getStatus(), (int) time);
        in.close();
        out.close();
        socket.close();
        return response;
    } catch (IOException e) {
        logger.severe( msg: "Error redirecting request: " + e.getMessage());
        return null;
    }

public HttpServletResponse requestAvailableServer(){
    logger.info( msg: "Requesting available server...");
    try {
        HttpResponseParser parser = new HttpResponseParser();
        Socket socket = new Socket(HttpServer.monitorAddress,HttpServer.monitorPort);
        DataOutputStream out = new DataOutputStream(socket.getOutputStream());
        out.writeUTF( str: "GET /getAvailableServer HTTP/1.1\r\n\r\n");
        socket.shutdownOutput();
        InputStream in = socket.getInputStream();
        HttpServletResponse response = parser.parse(in);

        socket.close();

        return response;
    } catch (IOException e) {
        logger.severe( msg: "Error requesting available server: " + e.getMessage());
        e.printStackTrace();
    }
    return null;
}
```

**Висновок:** Ми реалізували взаємодію програми в одній з архітектур відповідно до обраної теми