ENGI 9869 Advanced Concurrent Programming

A Presentation on
“MultiThreaded Client/Server Chat Application”

Inaam Ahmed

201692544

inaama@mun.ca

Foundation

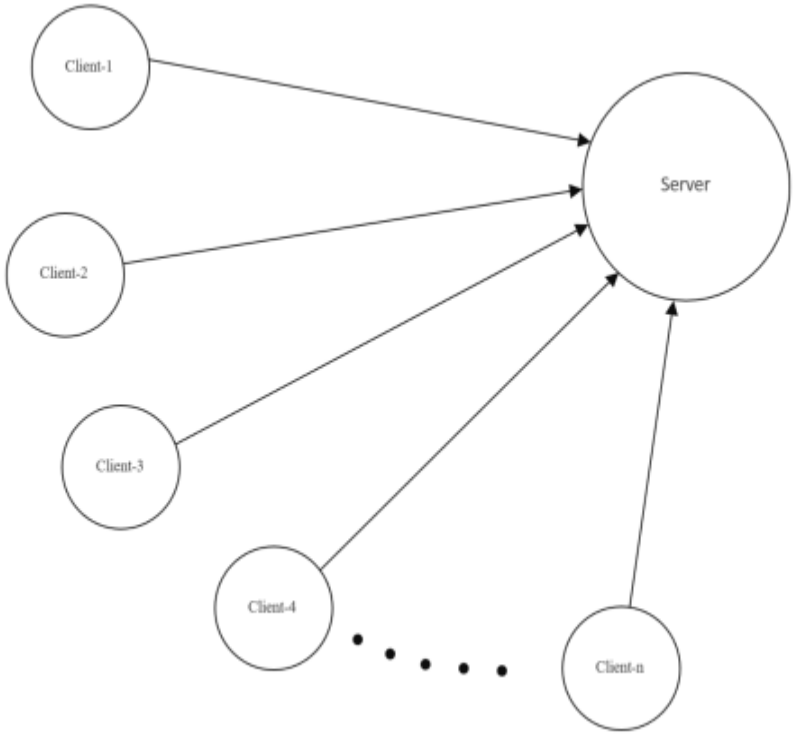
- Concurrent programming is among **advanced structural programming** technique
- Number of operations executed in same time by **context switching**
- These operations are literally called **Threads or processes**
- Concurrent processes have more than one **threads of control** not necessarily
- Limited set of resources **enforce synchronization**

Problem Statement View

- User identified his/her **Name**
- **All users can see** messages with sender identification
- **Notification leaveing** chat room
- Messages received will be logged by current **date and time** of arrival
- Necessary **synchronization** among the threads enforced
- One Server **running continuously** (Just in Time Availability) is required to run the application
- 10’s of users enforcing **concurrency**
- Host Machine is Intel(R) Core (TM) i7, 6700 HQ CPU @ **2 x 2.6 GHz**

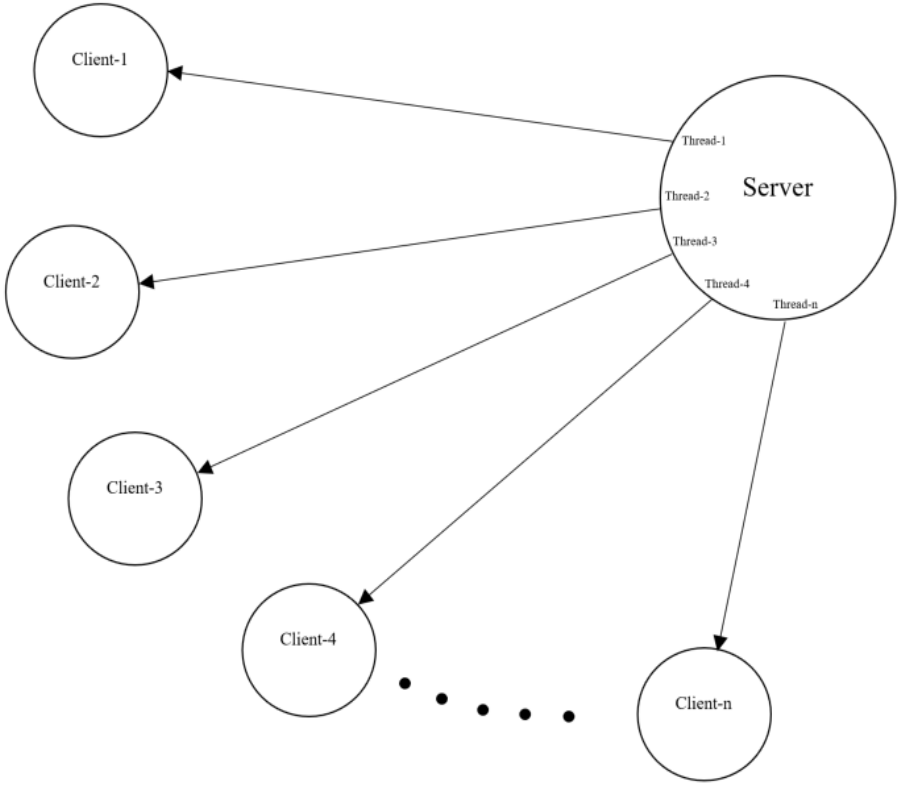
Application Design

- Client-Server Paradigm/Architecture
- Java multithreading which gives an **explicit implementation of threads** creation and control synchronization over the layer of Client Server Architecture



Application Design

- Server creates a **new thread in response to every client** and do all reads and writes and throwing received message to all the client threads currently running in on the server



Suggested Algorithm

Server

```
Begin
        Initiate server process waiting for requests
                Receive request
                Create a thread for client //fork
                Create Socket connection
                While Comm. Not Ends
                        Do input/output streams
                Close streams
                Close socket
                Terminate thread
End
```

Server will start receiving requests again after creating thread for requested client

Suggested Algorithm

Client

```
Begin
    Initiate client process
        Create a thread for making request
            Send Request to server
        Create client thread for Data Transfer
        While Comm. Not Ends
            Do input/output streams
        Close I/O streams
        Close socket
        Kill thread
End
```

Programming Granularity

- Sockets is one **end point of two way** communication link and complex data structures
- Higher levels of programming abstraction are also available like **interfaces of Java, RMI**
- Java network Sockets will provide **lower level connection orientation**
- **Input and output streams** used for data forwarding and receiving over socket connections
- **Machine_ID** and **port number** necessary to establish a connection with in machines using sockets.
- **Machine_ID** is identification number of specific machine **IP**
- **Port_No** is identification number of specific application.

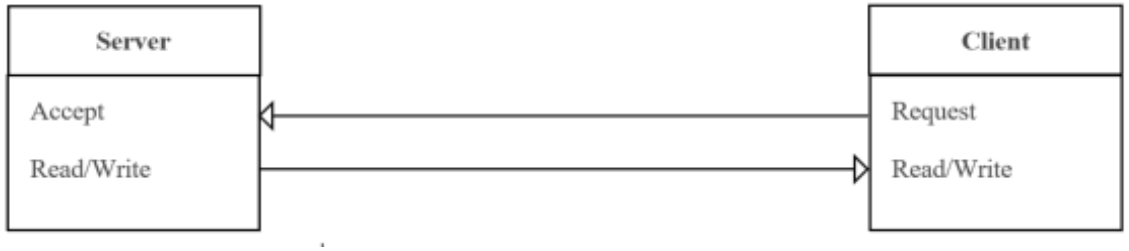
Programming Granularity

- **Java.net** is a package in Java development platform which contains **Internet Address resolutions, sockets** with **TCP** and **UDP** connections.
- Here we used TCP
  - Socket Class
  - Server Socket Class
- **Connection must be established** between client and server before start sending or receiving between client to server or vice versa

Implementing Sockets

- Implemented on **both server and client side** of application.
- TCP sockets are **connection oriented** sockets
- Use **Socket and Server Socket objects** for communication.

1. Create Network Connection
2. Open Sockets
3. Creating Input/ Output Streams //Data Transfer
4. Close Sockets



How to Run

- This project is implemented on Intel(R) Core (TM) i7, 6700 HQ CPU @ **2 x 2.6** GHz machine.
- Address of current machine is **localhost**. Which may be **169.254.x.x/16** which are default address when the machine is disconnected.
- When it is connected **localhost the request will resolve** the current address of machine.
- Port Number we used here is **55555** which any number between **1024-65535** on which this machine and application will accept the requests.

Launch Application

Server

javac "Server_filename".java
java "Server_filename"

Client

javac "Client_filename".java
java "Client_filename"
"Repeat the process for multiple clients"

.

Running Example

Command Prompt - java MultiThreadChatServer

```
C:\Users\inaam\acp_project\project>java MultiThreadChatServer
Usage: java MultiThreadChatServer <portNumber>
Now using port number=60000
```

Running Example

C:\Windows\System32\cmd.exe - java MultiThreadChatClient

```
C:\Users\inaam\acp_project\project>java MultiThreadChatClient
Usage: java MultiThreadChatClient <host> <portNumber>
Now using host=localhost, portNumber=60000
Enter your name.
Inaam


Hello Inaam to our chat room.
To leave enter acp_quit in a new line
```

C:\Windows\System32\cmd.exe - Java MultiThreadChatClient

```
C:\Users\inaam\acp_project\project>Java MultiThreadChatClient
Usage: java MultiThreadChatClient <host> <portNumber>
Now using host=localhost, portNumber=60000
Enter your name.
***


  A new user Inaam entered the chat room !!! ***
```

C:\Windows\System32\cmd.exe - java MultiThreadChatClient

```
C:\Users\inaam\acp_project\project>java MultiThreadChatClient
Usage: java MultiThreadChatClient <host> <portNumber>
Now using host=localhost, portNumber=60000
Enter your name.
***


  A new user Inaam entered the chat room !!! ***
```

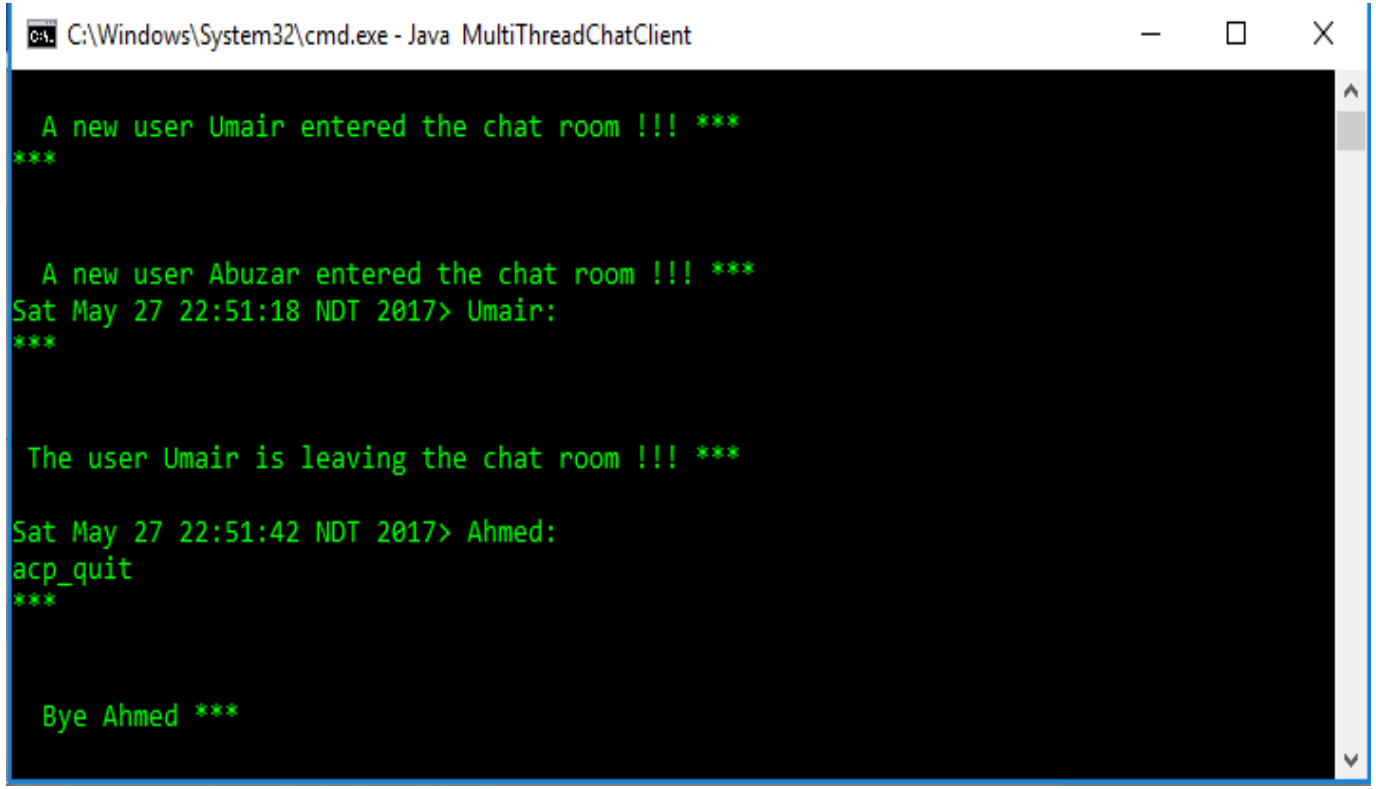
C:\Windows\System32\cmd.exe - Java MultiThreadChatClient

```
C:\Users\inaam\acp_project\project>Java MultiThreadChatClient
Usage: java MultiThreadChatClient <host> <portNumber>
Now using host=localhost, portNumber=60000
Enter your name.
***


  A new user Inaam entered the chat room !!! ***
```

Closing Application

To exit from chat group client must have to enter keyword “quit” and press enter he/she will eventually get out of the chat room.



Resource Monitor

File Monitor Help

Overview | CPU | Memory | Disk | Network

CPU

26% CPU Usage | 110% Maximum Frequency

Image	PID	Description	Status	Threads	CPU	Average CPU
macmnsvc.exe	2808	McAfee Agent Common Services	Running	7	0	0.00
lsass.exe	884	Local Security Authority Process	Running	8	0	0.00
jusched.exe	3696	Java Update Scheduler	Running	2	0	0.00
jhi_service.exe	5660	Intel(R) Dynamic Application Loader Host Interface	Running	2	0	0.00
java.exe	6092	Java(TM) Platform SE binary	Running	27	0	0.01
java.exe	7472	Java(TM) Platform SE binary	Running	25	0	0.01
java.exe	6904	Java(TM) Platform SE binary	Running	24	0	0.00
java.exe	10572	Java(TM) Platform SE binary	Running	27	0	0.00
java.exe	5056	Java(TM) Platform SE binary	Running	24	0	0.00

Disk

140 KB/sec Disk I/O | 0% Highest Active Time

Image	PID	File	Read (B/sec)	Write (B/sec)	Total (B/sec)	I/O Priority	Response Time ...
System	4	C:\Users\inaam\AppData\Local\TileDataLayer\Datab...	32,768	24,576	57,344	Normal	2
System	4	C:\Users\inaam\AppData\Local\Packages\Microsoft....	0	4,096	4,096	Normal	2
System	4	C:\Users\inaam\AppData\Local\TileDataLayer\Datab...	0	8,192	8,192	Normal	2
dllhost.exe	3548	C:\$LogFile (NTFS Volume Log)	0	12,288	12,288	Normal	2
System	4	C:\Users\inaam\AppData\Local\Packages\Microsoft....	0	6,144	6,144	Normal	2
System	4	C:\Users\inaam\AppData\Local\TileDataLayer\Datab...	0	8,192	8,192	Normal	2
SearchUI.exe	2536	C:\$LogFile (NTFS Volume Log)	0	22,528	22,528	Normal	1
System	4	C:\Windows\System32\LogFiles\WMI\Wifi.etl	0	2,731	2,731	Normal	1
System	4	C:\ProgramData\Microsoft\Windows\wfp\wfpdiag.etl	0	1,092	1,092	Normal	1

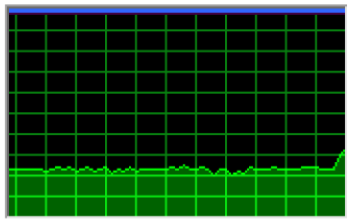
Network

108 Kbps Network I/O | 0% Network Utilization

Image	PID	Address	Send (B/sec)	Receive (B/sec)	Total (B/sec)
SearchUI.exe	2536	13.107.21.200	46,512	28,489	75,001
GROOVE.EXE	9044	40.117.151.29	45	242	287
OfficeClickToRun.exe	6440	40.117.151.29	27	148	175
System	4	13.107.21.200	100	0	100
java.exe	7472	LAPTOP-I6EPVB3A	23	1	24
svchost.exe (NetworkService)	1860	LAPTOP-I6EPVB3A.hitronhub.home	0	9	9

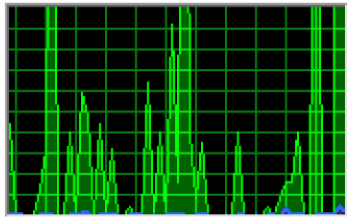
Views

CPU 100%



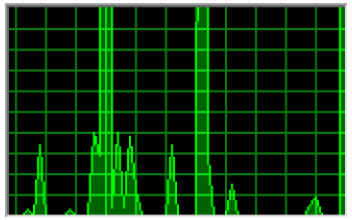
60 Seconds 0%

Disk 100 KB/sec



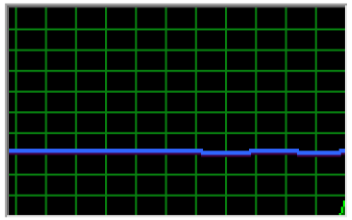
0

Network 10 Kbps



0

Memory 100 Hard Faults/sec



0

Conclusion

- The application is limited to **5-10** clients only.
- **No authentication** for a user to enter the chat room. AAA can be implemented with basic java structures but less concerned in this project
- Authentication is a more important measure for a **chat application**.
- As you can see we have Multithreaded Client-Server application with is **only textual communications** where no multimedia item can be shared
- Java **web resources** Upgradation (file sharing, searching media, tracing history etc.)
- The application can be enhanced to a **Graphical Interface**

Questions