

User Manual for P2P Network File Sharing System

1. Introduction

Welcome to the **Peer-to-Peer (P2P) Network File Sharing System** manual. This guide provides detailed, step-by-step instructions to help users set up, compile, run, and test the P2P file sharing application. The system leverages Java Remote Method Invocation (RMI) to enable seamless communication between SuperPeers and LeafNodes, ensuring efficient file distribution and consistency across the network.

2. Prerequisites

Before proceeding, ensure that the following requirements are met:

- **Operating System:** Windows 10 (or later) / Linux / macOS
- **Java Version:** Java SE 8 or higher
- **Development Environment:** Any Java IDE (e.g., Eclipse, IntelliJ IDEA) or command-line tools
- **Network Configuration:** Localhost setup with multiple ports to simulate different peers and superpeers
- **Required JAR Files:**
 - Gnutella_P2P.jar
 - javax.ws.rs-api-2.0.jar
- **Directory Structure:**
 - Source Code Directory: C:\Users\dattu\CS485_PA3
 - Libraries Directory: C:\Users\dattu\CS485_PA3\lib
 - Binary Output Directory: C:\Users\dattu\CS485_PA3\bin
 - Source Files Location: C:\Users\dattu\CS485_PA3\src\com\gfiletransfer

3. Setup Instructions

3.1. Navigate to the Source Directory

1. Open Command Prompt:

- Press Win + R, type cmd, and press Enter.

2. Change Directory:

```
cd C:\Users\dattu\CS485_PA3
```

3.2. Compile the Source Code

1. Ensure Required JARs are Present:

- Verify that Gnutella_P2P.jar and javax.ws.rs-api-2.0.jar are located in the lib directory.

2. Compile the Java Source Files:

```
javac -cp lib/javax.ws.rs-api-2.1.1.jar -d bin src/com/gfiletransfer/*.java
```

○ Explanation:

- -cp lib/javax.ws.rs-api-2.1.1.jar: Sets the classpath to include the required JAR file.
- -d bin: Specifies the destination directory for compiled .class files.
- src/com/gfiletransfer/*.java: Compiles all Java source files in the specified package.

3. Verify Compilation:

- Ensure that the bin directory now contains the compiled .class files without any compilation errors.

3.3. Start the RMI Registry

1. Open a New Command Prompt Window:

- Press Win + R, type cmd, and press Enter.

2. Navigate to the Bin Directory:

```
bash
```

```
cd C:\Users\dattu\CS485_PA3\bin
```

3. Start RMI Registry:

```
start rmiregistry
```

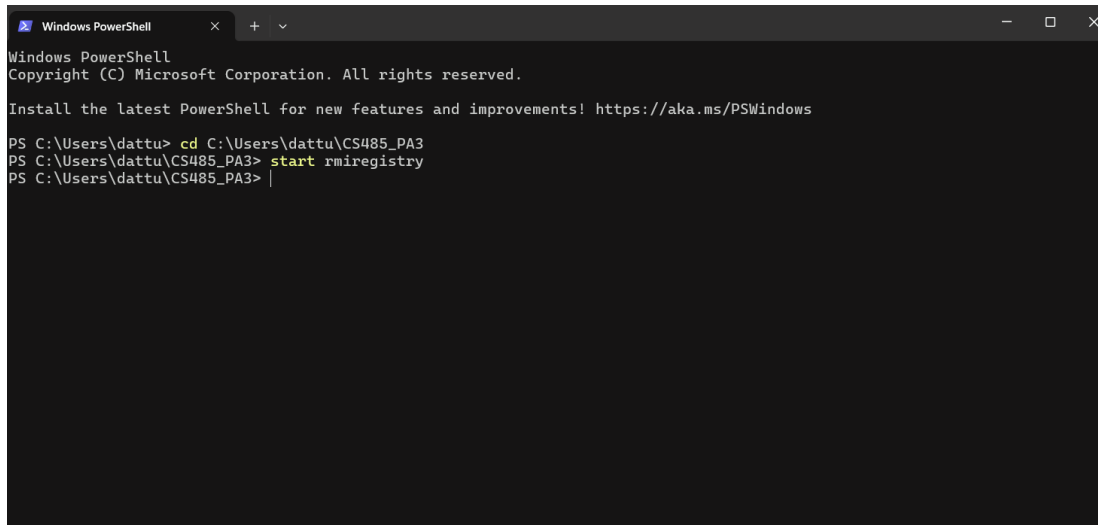
- **Note:** This command starts the RMI registry in the background. Ensure that it remains running while the SuperPeer and LeafNodes are active.

3.4. Running the Server and Peers

The system includes three executable batch files:

1. **run_server.bat**
2. **run_peer.bat**
3. **run_test.bat**

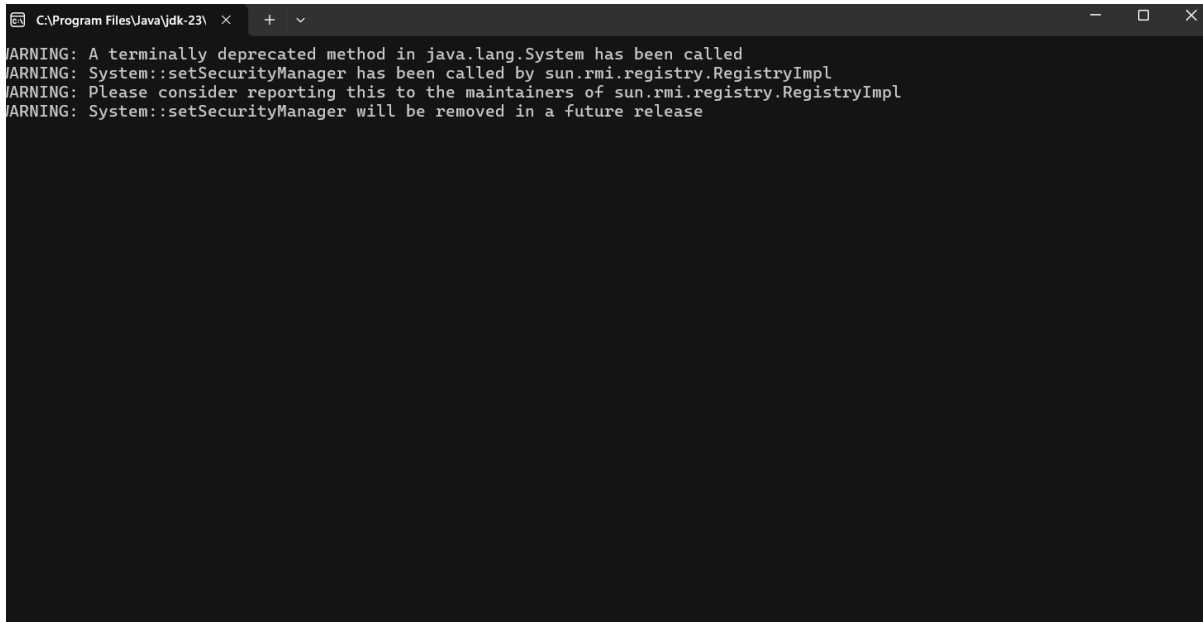
Important: These batch files will run only if the corresponding .jar files (Gnutella_P2P.jar and javax.ws.rs-api-2.0.jar) are present in the lib directory.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\dattu> cd C:\Users\dattu\CS485_PA3
PS C:\Users\dattu\CS485_PA3> start rmiregistry
PS C:\Users\dattu\CS485_PA3> |
```

A screenshot of a Java IDE's console window. The title bar shows the file path 'C:\Program Files\Java\jdk-23\'. The console output displays three deprecation warnings from the Java runtime. The first warning states that a terminally deprecated method in java.lang.System has been called. The second warning indicates that System::setSecurityManager has been called by sun.rmi.registry.RegistryImpl. The third warning asks the user to consider reporting this to the maintainers of sun.rmi.registry.RegistryImpl and notes that System::setSecurityManager will be removed in a future release.

```
WARNING: A terminally deprecated method in java.lang.System has been called
WARNING: System::setSecurityManager has been called by sun.rmi.registry.RegistryImpl
WARNING: Please consider reporting this to the maintainers of sun.rmi.registry.RegistryImpl
WARNING: System::setSecurityManager will be removed in a future release
```

4. Step-by-Step Test Case: PUSH-Based Consistency

This test case demonstrates the **PUSH-based** consistency mechanism by simulating file modifications and ensuring that invalidation messages are correctly propagated to cached copies across the network.

Step 1: Start the SuperPeer and LeafNodes

Objective: Initialize the SuperPeer and multiple LeafNodes, ensuring they are correctly registered and running.

Super Peer Console (run_server.bat for SP01):

1. Run the SuperPeer:

- Navigate to the bin directory:

```
cd C:\Users\dattu\CS485_PA3\bin
```

- Execute the SuperPeer batch file:

```
run_server.bat
```

```

"SuperPeer is now Running. Please configure your settings."
"#####"
Enter the Super Peer ID.
sp01
Loaded 10 peer configurations.
Loaded 10 superpeer configurations.
Raw Line: #All_To_All_Topology and Linear Topology Properties : Peer_ID; Linear_Neighbour; All_Neighbours
Raw Line: SP01;SP02;SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP09,SP10
Tokens: [SP01, SP02, SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP09,SP10]
Raw Line: SP02;SP03;SP01,SP03,SP04,SP05,SP06,SP07,SP08,SP09,SP10
Tokens: [SP02, SP03, SP01,SP03,SP04,SP05,SP06,SP07,SP08,SP09,SP10]
Raw Line: SP03;SP04;SP01,SP02,SP04,SP05,SP06,SP07,SP08,SP09,SP10
Tokens: [SP03, SP04, SP01,SP02,SP04,SP05,SP06,SP07,SP08,SP09,SP10]
Raw Line: SP04;SP05;SP01,SP02,SP03,SP05,SP06,SP07,SP08,SP09,SP10
Tokens: [SP04, SP05, SP01,SP02,SP03,SP05,SP06,SP07,SP08,SP09,SP10]
Raw Line: SP05;SP06;SP01,SP02,SP03,SP04,SP06,SP07,SP08,SP09,SP10
Tokens: [SP05, SP06, SP01,SP02,SP03,SP04,SP06,SP07,SP08,SP09,SP10]
Raw Line: SP06;SP07;SP01,SP02,SP03,SP04,SP05,SP07,SP08,SP09,SP10
Tokens: [SP06, SP07, SP01,SP02,SP03,SP04,SP05,SP07,SP08,SP09,SP10]
Raw Line: SP07;SP08;SP01,SP02,SP03,SP04,SP05,SP06,SP08,SP09,SP10
Tokens: [SP07, SP08, SP01,SP02,SP03,SP04,SP05,SP06,SP08,SP09,SP10]
Raw Line: SP08;SP09;SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP09,SP10
Tokens: [SP08, SP09, SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP09,SP10]
Raw Line: SP09;SP10;SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP10
Tokens: [SP09, SP10, SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP10]
Raw Line: SP10;SP01;SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP09
Tokens: [SP10, SP01, SP01,SP02,SP03,SP04,SP05,SP06,SP07,SP08,SP09]
Loaded 10 topology configurations.
Topology type: ALL
Consistency mechanism: PUSH

```

Instructions:

When prompted, enter the SuperPeer ID (e.g., SP01).

The SuperPeer will load configurations and start running, listening on the port specified in the property file.

Start Additional SuperPeers (SP02, SP03):

Open additional Command Prompt windows for each SuperPeer.

Repeat the above steps, entering unique SuperPeer IDs (e.g., SP02, SP03).

Leaf Node Console (run_peer.bat for P01, P02, P03):

1. Run a LeafNode:

- Navigate to the bin directory:

```
cd C:\Users\dattu\CS485_PA3\bin
```

- Execute the LeafNode batch file:

```
run_peer.bat
```

2. Configure LeafNode:

- **Console Output for P01:**

```

PS C:\Users\dattu> cd C:\Users\dattu\CS485_PA3
PS C:\Users\dattu\CS485_PA3> ./run_peer.bat
"Setting the Class path"
"Running the Peer..."

Waiting for 2 seconds, press a key to continue ...
"#####"
"Peer is now Running. Please configure your settings."
"#####"
Enter Peer ID
p01
Loaded 10 peer configurations.

Registering details of File name file1.txt in Indexing Server
Registering details of File name file10.txt in Indexing Server
Registering details of File name file2.txt in Indexing Server
Registering details of File name file3.txt in Indexing Server
Registering details of File name file4.txt in Indexing Server
Registering details of File name file5.txt in Indexing Server
Registering details of File name file6.txt in Indexing Server
Registering details of File name file7.txt in Indexing Server
Registering details of File name file8.txt in Indexing Server
Registering details of File name file9.txt in Indexing Server
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)

```

1.

- **Instructions:**

- When prompted, enter the Peer ID (e.g., p01).
- The LeafNode will load configurations, register master files with the SuperPeer, and start running.

2. **Start Additional LeafNodes (p02, p03):**

- Open additional Command Prompt windows for each LeafNode.
- Repeat the above steps, entering unique Peer IDs (e.g., p02, p03).

Step 2: Modify a File (On P01 to Trigger the Push Mechanism)

Objective: Edit a master file on LeafNode p01 to trigger the PUSH-based invalidation mechanism, ensuring that other peers are notified of the update.

```
Enter Peer ID: p01
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
edit
Enter the file name which you want to Edit(Append): file1.txt
Enter anything you want to append in this file: Good day, this is version 2.
File edited Successfully.
New Version Number for edited file : v02
Send invalidate request to all nodes (PUSH)
```

Instructions:

- At the prompt, type edit and press Enter.
- Enter the filename to edit (e.g., file1.txt).
- Append the desired content (e.g., Good day, this is version 2.).
- The system confirms the edit and sends invalidation requests to all cached copies.

Step 3: Invalidating the Cached Files on Other Peers (Push Mechanism)

Super Peer Console (SP01):

Objective: Observe how cached copies on other LeafNodes (p02, p03) receive invalidation messages and update their cached tables accordingly.

Broadcasting Invalidate Request for file1.txt to all peers

1. The SuperPeer (SP01) automatically broadcasts the invalidation request upon receiving the edit command from p01.

Leaf Node Consoles (p02, p03):

```
Enter Peer ID: p03
Loaded 10 peer configurations.
Loaded 10 superpeer configurations.
Topology type: ALL
Consistency mechanism: PUSH
Updated Cached Table Entry after Invalidation from Push
file1.txt => [file1.txt, CC, cached_dir_P03, v01, p03, invalid, 30]
```

Instructions:

- Upon receiving the invalidation request, each LeafNode updates its cached table, marking file1.txt as invalid.

Step 4: Peer Downloads the Updated Version

Objective: Ensure that LeafNodes (p02, p03) download the updated version of file1.txt from the master copy (p01) after invalidation.

Leaf Node Consoles (p02, p03):

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Now Started Calling the query() from Leaf Node...
#####
VALID Peer providing the file with Peer ID is p01 under Super Peer :SP01 which is a Master Copy
#####
Enter Peer ID you wish to take the file from: p01
File Downloading Successful.
Display File file1.txt
Updated Cached Table Entry after insertion (File download)
file1.txt => [file1.txt, CC, cached_dir_P02, v01, p01, valid, 30]
```

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Now Started Calling the query() from Leaf Node...
#####
VALID Peer providing the file with Peer ID is p01 under Super Peer :SP01 which is a Master Copy
#####
Enter Peer ID you wish to take the file from: p01
File Downloading Successful.
Display File file1.txt
Updated Cached Table Entry after insertion (File download)
file1.txt => [file1.txt, CC, cached_dir_P03, v02, p01, valid, 30]
```

Instructions:

- At the prompt, type search and press Enter.
- Enter the filename to search (e.g., file1.txt).
- The system displays peers holding the master copy.
- Enter the Peer ID of the master copy (p01) to download the updated file.
- The system confirms the successful download and updates the cached table with the new version (v02) and marks it as valid.

Step 5: Peer P01 Verifies that All Peers Are Updated

Objective: Confirm that all LeafNodes (p01, p02, p03) have the latest version of file1.txt, ensuring consistency across the network.


```

Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Now Started Calling the query() from Leaf Node...
#####
VALID Peer providing the file with Peer ID is p01 under Super Peer :SP01 which is a Master Copy
VALID Peer providing the file with Peer ID is p02 under Super Peer :SP01 which is a Master Copy
VALID Peer providing the file with Peer ID is p03 under Super Peer :SP01 which is a Master Copy
#####
Enter Peer ID you wish to take the file from: p02
File Downloading Successful.
Display File file1.txt
Updated Cached Table Entry after insertion (File download)
file1.txt => [file1.txt, CC, cached_dir_P01, v02, p02, valid, 30]

```

Instructions:

- At the prompt, type search and press Enter.
- Enter the filename to search (e.g., file1.txt).
- The system lists all peers holding the master and cached copies of the file.
- Enter the Peer ID of another LeafNode (p02) to download the file.
- The system confirms the successful download and updates the cached table accordingly.

Step 6: End the Process

Objective: Gracefully terminate the LeafNode and SuperPeer applications after completing the test.

```

Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
exit
Press any key to continue . . . . .

```

Instructions:

At the prompt, type exit and press Enter.

Press any key to confirm termination.

Super Peer Console (SP01):

Broadcast Invalidate Request for file1.txt completed successfully.

Instructions:

- Close the SuperPeer console or terminate the application as needed.

Step-by-Step Test Case: PULL-Based Consistency

This test case demonstrates the PULL-based consistency mechanism by simulating file modifications and observing how cached copies periodically poll the origin server to maintain consistency.

Step 1: Peer Downloads Files from P01 (Master Copy)

Objective: LeafNodes (p02, p03) download file1.txt from the master copy held by p01, creating cached copies that will be subject to polling.

Leaf Node Consoles (p02, p03):

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Now Started Calling the query() from Leaf Node...
#####
VALID Peer providing the file with Peer ID is p01 under Super Peer :SP01 which is a Master Copy
#####
Enter Peer ID you wish to take the file from: p01
File Downloading Successful.
Display File file1.txt
Updated Cached Table Entry after insertion (File download)
file1.txt => [file1.txt, CC, cached_dir_P02, v01, p01, valid, 30]
```

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Now Started Calling the query() from Leaf Node...
#####
VALID Peer providing the file with Peer ID is p01 under Super Peer :SP01 which is a Master Copy
#####
Enter Peer ID you wish to take the file from: p01
File Downloading Successful.
Display File file1.txt
Updated Cached Table Entry after insertion (File download)
file1.txt => [file1.txt, CC, cached_dir_P03, v02, p01, valid, 30]
```

Instructions:

- At the prompt, type search and press Enter.
- Enter file1.txt as the filename to search.
- Select p01 as the Peer ID to download the file from.
- The system confirms the successful download and updates the cached table.

Step 2: P01 Edits the File

Objective: The origin server (p01) modifies file1.txt, potentially causing cached copies to become outdated.

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
edit
Enter the file name which you want to Edit(Append): file1.txt
Enter anything you want to append in this file: Good day, this is version 2 PULL.
File edited Successfully.
New Version Number for edited file : v02
```

We have modified the file and it goes for invalid

Objective: Now that the Time-To-Refresh (TTR) for the downloaded file file1.txt has expired, p02 and p03 will poll the origin server to check if the cached copy is still valid.

Upon receiving the invalidation request, each LeafNode updates its cached table, marking file1.txt as invalid.

Polling and TTR Expiry for P02 and P03

```
Enter the file name which you want to search: file1.txt
Polling for file1.txt, checking for TTR expiration...
Polling initiated for file: file1.txt
Polling response from origin server: File is out of date.
Marking file1.txt as invalid due to expired TTR and modification on origin server.
Updated Cached Table Entry after Invalidation
file1.txt => [file1.txt, CC, cached_dir_P02, v01, p01, invalid, 30]
```

```
Do you want to Search a File, Delete File, Edit file or Exit? (Search/Delete/Edit/Exit)
search
Enter the file name which you want to search: file1.txt
Polling for file1.txt, checking for TTR expiration...
Polling initiated for file: file1.txt
Polling response from origin server: File is out of date.
Marking file1.txt as invalid due to expired TTR and modification on origin server.
Updated Cached Table Entry after Invalidation
file1.txt => [file1.txt, CC, cached_dir_P03, v01, p01, invalid, 30]
```

Instructions:

- After the configured TTR period (e.g., 30 seconds), the system automatically initiates polling.
- The console displays the polling process and updates the cached table upon detecting the outdated file.

Push: Querying Node (p01), Modifying: p02, p03

Percentage of Invalid query results: 2 and % 6.70

Similarly done for p02 and p03

Pull

PULL, TTR = 180 seconds, Querying: p01, p02, p03; Modifying: p04, p05, p06

Percentage of Invalid query results: 8 and % 13.33

Similarly done 30s,60s,120s.

-----End-----