Programming Assignment 3

Verification Document

Maintaining file consistency a Gnutella-styleP2P system

Arun Mathew Iype	A20278285
Maheshwara Reddy	A20284393

Contents

Component Testing

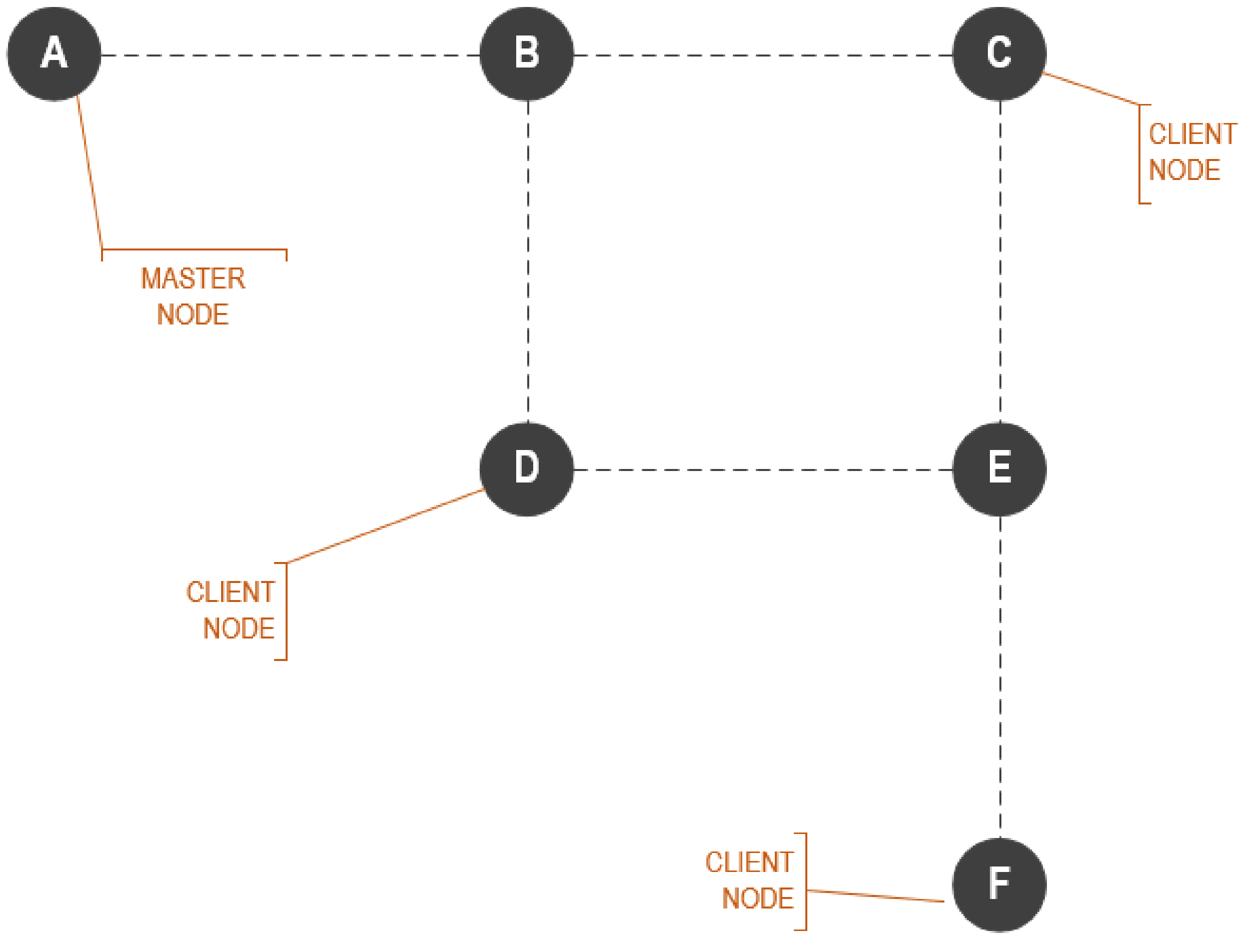
Test	**Test Steps**	**Expected Result**	**Actual result**
Message Transfer	1) Create a main() to run the test for message transfer function. 2) Create a main() to run a function to accept and display all received messages. 3) Start the receive test program. 4) Start the send test program.	The message sent by the send test program is received by the receive test program.	The message was correctly received by the receive test program.
File Transfer	1) Create a main() to run the file accept part of the file transfer function. 2) Create a main() to run the send part of the file transfer function. 3) Run the file receive test program. 4) Run the file send test program.	The file should be received by the receive test program.	The file was successfully transferred.
Periodic Housekeeping tasks	1) Create a main() to run the housekeeping function. 2) Run the test program.	The test program should display a set of messages on the screen to show the thread processing.	The test messages are seen as expected.
Detect File Modifications	1) Create a main() to run/start the function-thread to detect any modifications to files 2) Start the test.	The function-thread should detect changes to the files in the "Source" directory and display them on the screen.	The function can detect the changes to files in the Source Directory and display them on the screen.

Integration Testing

Test	Test Steps	Expected Result	Actual result
Search File	1) Start peer 1 2) Start peer 2 3) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 1kb	1) Create/copy file Test1.txt of size 1KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 2kb	1) Create/copy file Test1.txt of size 2KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 3kb	1) Create/copy file Test1.txt of size 3KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 5kb	1) Create/copy file Test1.txt of size 5KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 6kb	1) Create/copy file Test1.txt of size 6KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
Obtain file of size 10kb	1) Create/copy file Test1.txt of size 10KB in peer 1 working directory 2) Start peer 1 3) Start peer 2 4) Enter “Test1.txt” in peer2	File Test1.txt with the same content is created in peer2	Success
For the following tests the Nodes setup shown in the figure below is used. The Node "A" is the Master Node for the file "test_6kb.txt", and initially none of the other Nodes have this file.			
Obtain file test_6kb.txt at Node "C"	1) Search for the file test_6kb.txt at Node "C".	1) The Node "C" should show only Node "A" in its search results	Success
	2) Select Node "A" to Obtain the File.	2) File should be retrieved from Node	Success

		"A".	
Obtain file test_6kb.txt at Node "D"	3) Search for the file test_6kb.txt at Node "D".	3) The Node "D" should show Nodes "A" and C" in its search results.	Success
	4) Select Node "C" to Obtain the File.	4) File should be retrieved from Node "C".	Success
Modify file at Node "A"	5) Open the file test_6kb.txt at Node "A" and change it. 6) After 5) search for test_6kb.txt at Node "F"	5,6) Node "F" should show only "A" as the source of the file.	Success
	7) Select Node "A" to Obtain the File.	7) File should be retrieved from Node "A".	Success

This test setup is also used for performance testing. The Node "A" is the Master Node for the file "test_6kb.txt". The Nodes "C", "D" and "F" are programmed to repeatedly search for the file after a particular interval.


A
B
C
CLIENT NODE
MASTER NODE
D
E
CLIENT NODE
CLIENT NODE
F