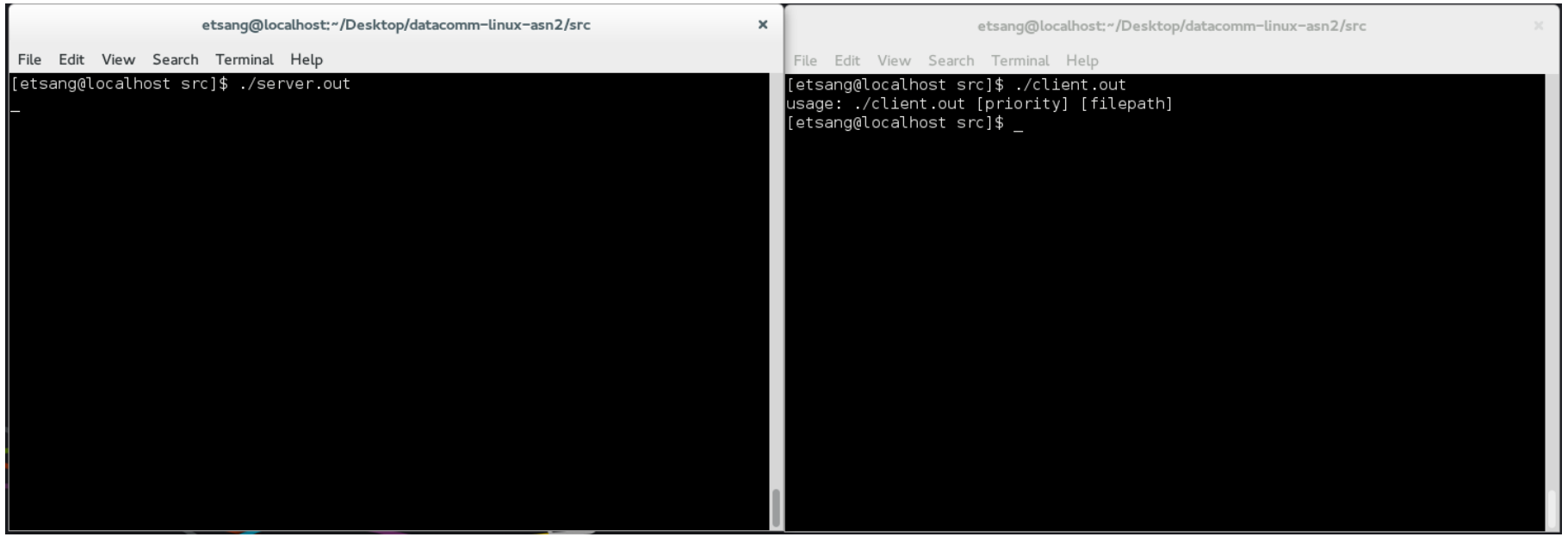

Figure 1bad command line arguments

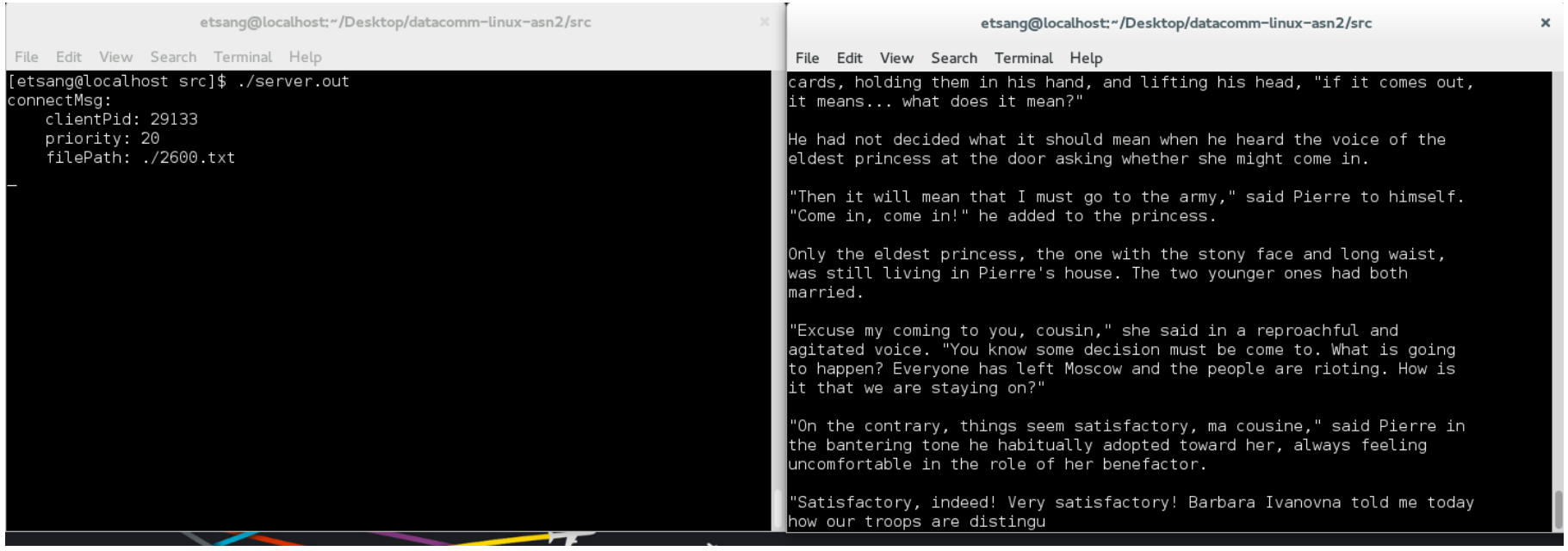

Figure 2it works with good command line arguments

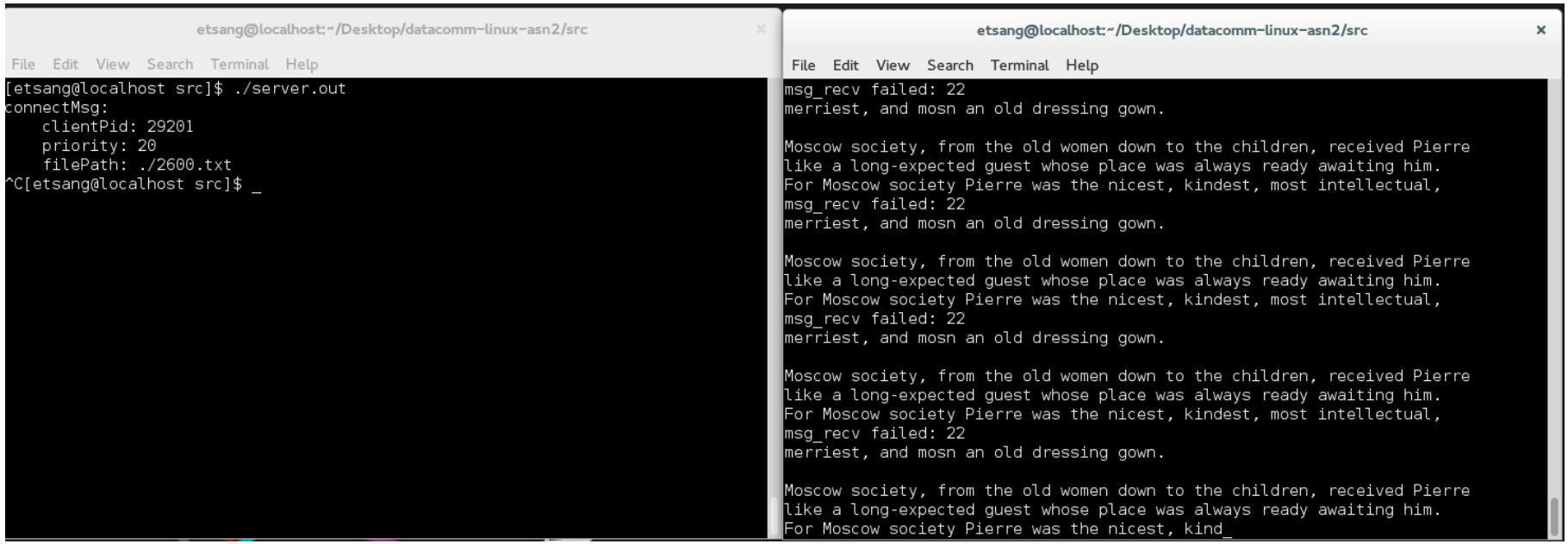

Figure 3[ctrl+c] on server causes issue on client

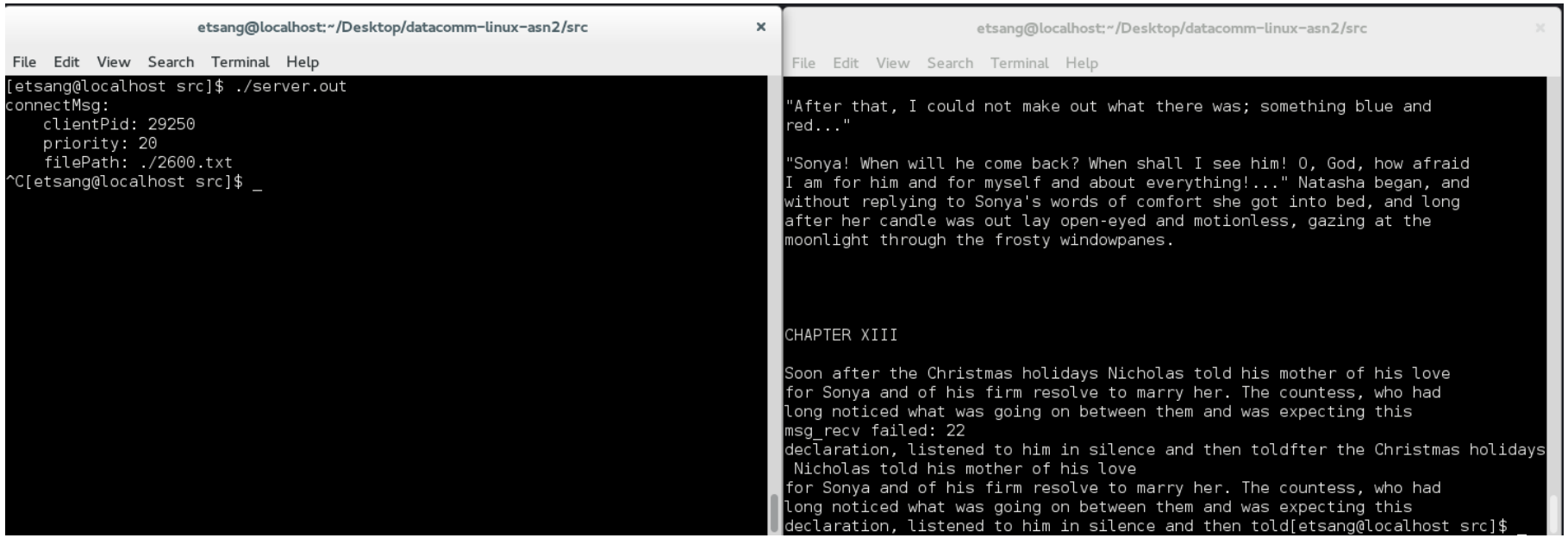

Figure 4[ctrl+c] on server no longer makes client go crazy

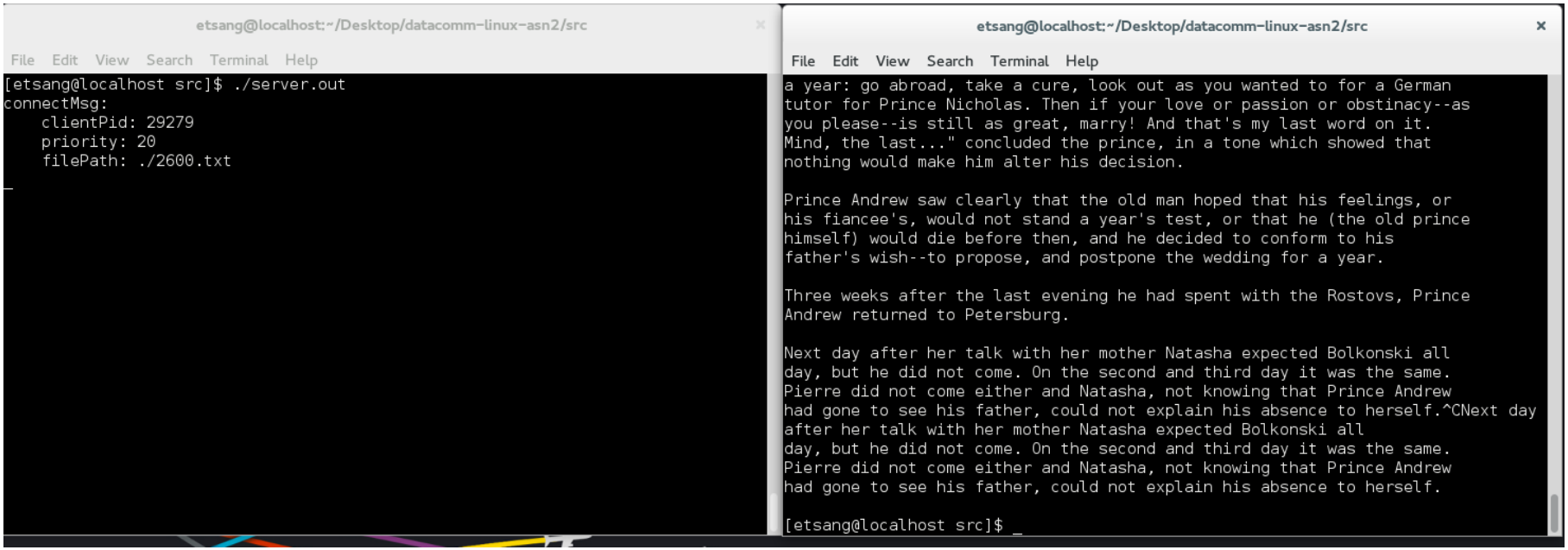

Figure 5[ctrl+c] on client freezes server

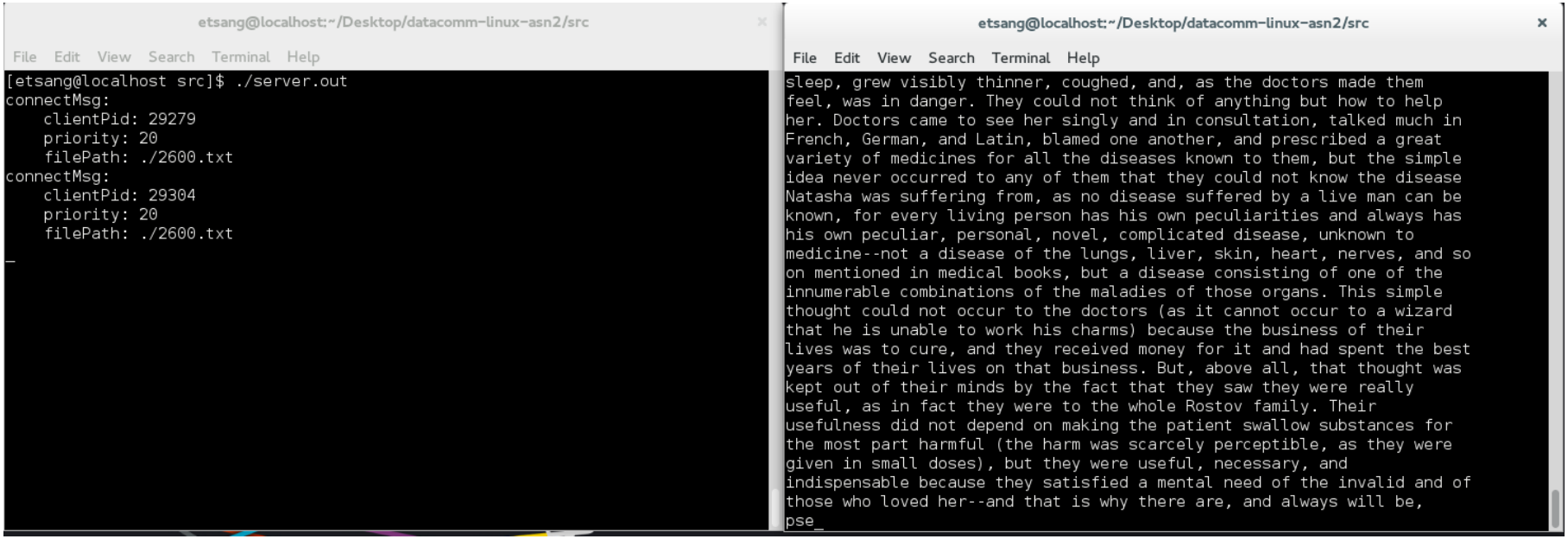

Figure 6[ctrl+c] on client no longer freezes server

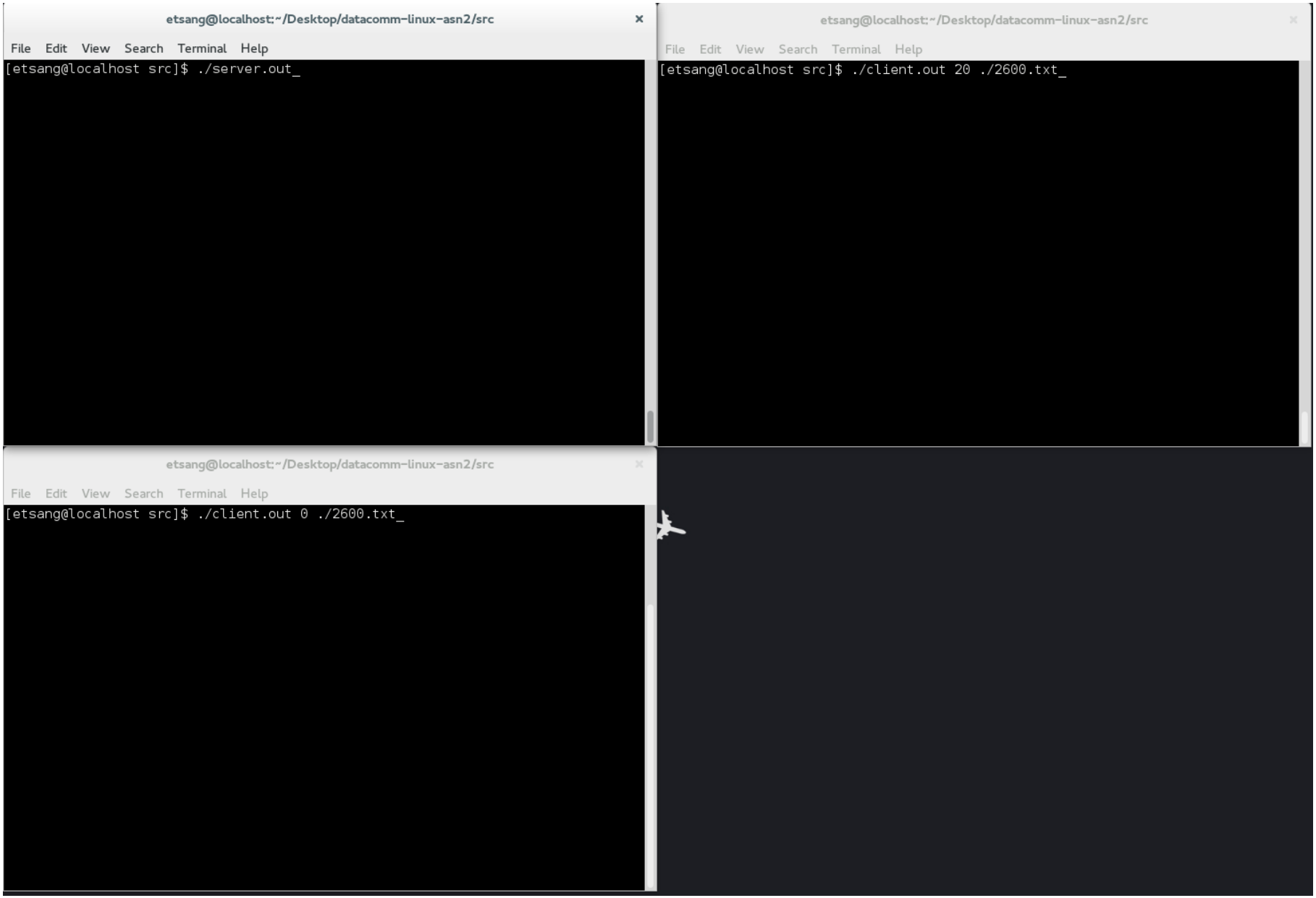

Figure 7before priority test

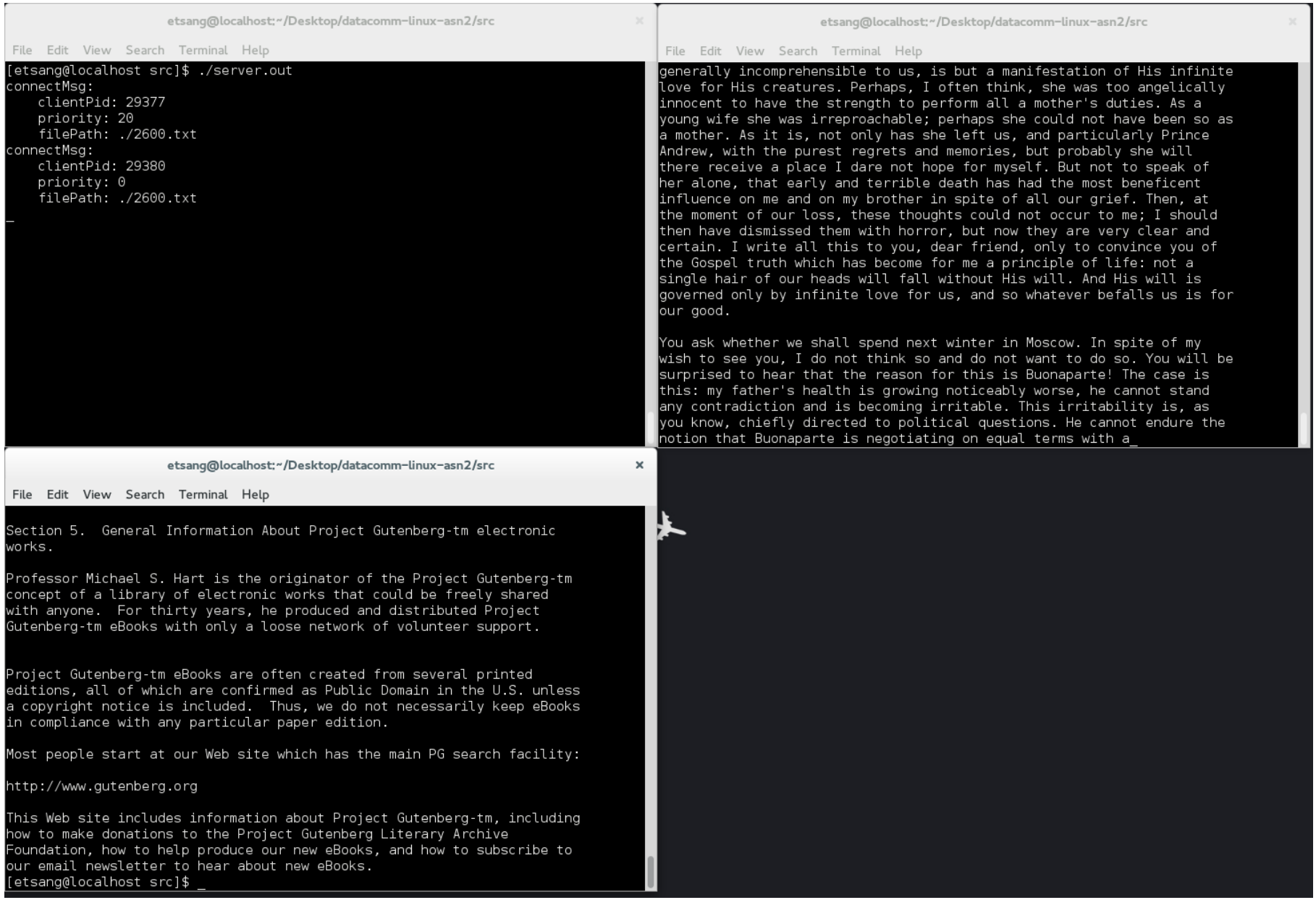

Figure 8priority 0 finishes first, as expected

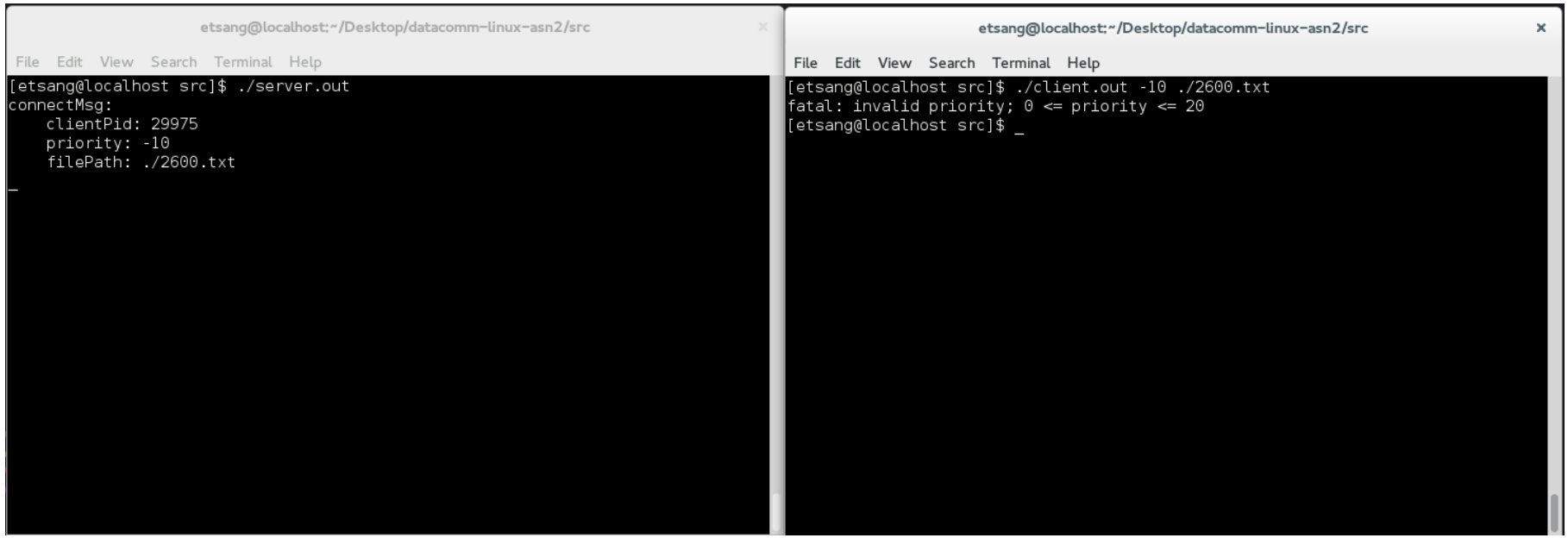

Figure 9priority error checking is fixed

Test #	Test procedure / description	Expected Outcome	Actual Outcome & Remarks	Pass / Fail
0	1. Start the server with "./server.out" 2. Start the client with "./client.out"	Server & client should start	Same as expected; I forgot to provide command line arguments to the client...the usage message reminded me	Pass; figure 1
1	1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt"	Server and client should start, and the contents of 2600.txt should scroll on the screen	Same as expected	Pass; figure 2
2	1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt" 3. do [ctrl+c] on the server	Server and client should start, and the contents of 2600.txt should scroll on the screen When [ctrl+c] is pressed, the client should exit, and so should the server	Server ended, and removed its message queue, but client kept printing the last message it got over and over again; see bug 0	Fail; figure 3
3	Same as above	Same as above	Same as expected	Pass; figure 4
4	1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt" 3. do [ctrl+c] on the client	Server and client should start, and the contents of 2600.txt should scroll on the screen When [ctrl+c] is pressed, the client should exit, but the server should not.	Same as expected, but i found a bug; see bug 1	Yes; figure 5
5	Procedure from bug 1	Second client should display text scrolling when executed	Same as expected	Yes; figure 6
6	1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt" 3. Start the another client with "./client.out	The client process started with priority 20 should finish later even though it was started first	Same as expected	Yes; figure 7 & 8

Test #	Test procedure / description	Expected Outcome	Actual Outcome & Remarks	Pass / Fail
	0 ./2600.txt"			
7	1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt" 3. Start the another client with "./client.out -20 ./2600.txt"	The client process started with priority -20 should finish first even though it was started last	Got an unexpected "permission denied" message; see bug 3	Fail
8	Same as test 7	The client process should print out that the priority range should only be between 0 or 20, inclusive	Same as expected	Pass; figure 9

Bug #	Steps to reproduce & effect	Cause	How it was resolved	Solved
0	**Procedures** Test 2 **Effect** Test 2	When reading from the message queue, I didn't check if it succeeds or not. so the loop just keeps printing the last received message over and over again.	Added error checking to the message receive call, and break out of the loop if an error occurs.	Yes; test 3
1	**Procedures** 1. Start the server with "./server.out" 2. Start the client with "./client.out 20 ./2600.txt" 3. do [ctrl+c] on the client 4. Start another client with "./client.out 20 ./2600.txt" **Effect** The second client process doesn't print anything onto the screen	The server process locks up, because the session processes fill up the message queue, and then nothing can be written into the message queue since there is no client to read from it.	Session processes communicate their PID to the client processes, so when the client terminates unexpectedly, they signal the session to clear the message queue, and terminate as well.	Yes; test 5
2	**Procedures** Test 7 **Effect** Test 7	User starting the processes didn't have the priority to give such high priorities.	Added validation of priority input, so it could only range from 0 to 20 instead of -20 to 20.	Yes; test 6 & test 8