WRITE USE CASE

| | | |
|---|---|---|
| Use Case Name | Write to Server | |
| Brief Description | User can write a file from the host to the server. | |
| Precondition | Server is running. | |
| Primary Actor | File Transfer System user | |
| Secondary Actor | None | |
| Dependencies to other use cases | None | |
| Basic flow | Steps | |
| | 1 | User inputs request type “write” into Client UI |
| | 2 | User inputs file path into Client UI |
| | 3 | User inputs data type “octet” or “netascii” into Client UI |
| | 4 | User inputs output mode “quiet” or “verbose” into Client UI |
| | 5 | User inputs mode “normal” or “test” into Client UI |
| | 6 | Client creates WRQ packet and sends it to the Server |
| | 7 | Server receives packet, parses WRQ, and spawns new ServerThread |
| | 8 | ServerThread creates an ACK packet and sends it to the client |
| | 9 | Client receives ACK packet |
| | 10 | DO:<br>Client creates & sends DATA packet to ServerThread.<br>ServerThread receives DATA packet.<br>ServerThread creates & sends ACK packet to Client.<br>Client receives ACK packet.<br>LOOP UNTIL: size of DATA packet data < 512 bytes |
| | 11 | UI prompts user for next input. |
| Global Alternative Flow | If server operator enters “shutdown” into the terminal THEN | |
| | 1 | Shutdown Server |
| | 2 | Abort |
| Bounded Alternative Flow | Step 7-10: IF opcode or filename or mode is invalid THEN | |
| | 1 | Server creates & sends ERR packet (code 4) |
| | 2 | Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file not found THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 1) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file access denied THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 2) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF insufficient disk space remaining THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 3) |

| | | |
|---|---|---|
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file already exists THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 6) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 9-10 | |
| | 1 | Client/Server receives ERR packet |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Specific Alternative Flow | Step 10 | |
| | 1 | DO:<br>Client does NOT receive ACK packet<br>Client waits 5 sec<br>Client re-sends DATA packet<br>LOOP UNTIL: timeout limit reached |
| | 2 | Abort |
| Specific Alternative Flow | Step 10: IF DATA packet TID is invalid THEN | |
| | 1 | Client creates & sends ERR packet (code 5) to the invalid TID |
| | 2 | Continue transfer |
| Specific Alternative Flow | Step 10: IF DATA packet TID is invalid THEN | |
| | 1 | Client creates & sends ERR packet (code 5) to the invalid TID |
| | 2 | Continue transfer |

READ USE CASE

| | | |
|---|---|---|
| Use Case Name | Read from Server | |
| Brief Description | User can read a file from the server. | |
| Precondition | Server is running. | |
| Primary Actor | File Transfer System user | |
| Secondary Actor | None | |
| Dependencies to other use cases | None | |
| Basic flow | Steps | |
| | 1 | User inputs request type "read" into Client UI |
| | 2 | User inputs file path into Client UI |
| | 3 | User inputs data type "octet" or "netascii" into Client UI |
| | 4 | User inputs output mode "quiet" or "verbose" into Client UI |
| | 5 | User inputs mode "normal" or "test" into Client UI |
| | 6 | Client creates RRQ packet and sends it to the Server |
| | 7 | Server receives packet, parses RRQ, and spawns new ServerThread |

| | | |
|---|---|---|
| | 8 | DO:<br>ServerThread creates & sends DATA packet to Client<br>Client receives DATA packet<br>Client creates & sends ACK packet to ServerThread.<br>ServerThread receives ACK packet<br>LOOP UNTIL: size of DATA packet data < 512 bytes |
| | 9 | Client UI prompts user for next input. |
| Global Alternative Flow | If server operator enters “shutdown” into the terminal THEN | |
| | 1 | Shutdown Server |
| | 2 | Abort |
| Bounded Alternative Flow | Step 7-10: IF opcode or filename or mode is invalid THEN | |
| | 1 | Server creates & sends ERR packet (code 4) |
| | 2 | Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file not found THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 1) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file access denied THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 2) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF insufficient disk space remaining THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 3) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Bounded Alternative Flow | Step 7-10: IF file already exists THEN | |
| | 1 | Client/Server creates & sends ERR packet (code 6) |
| | 2 | Client/Server terminates connection |
| | 3 | Abort |
| Specific Alternative Flow | Step 8 | |
| | 1 | DO:<br>ServerThread does NOT receive ACK packet<br>ServerThread waits 5 sec and times out<br>ServerThread re-sends DATA packet<br>LOOP UNTIL: timeout limit reached |
| | 2 | Abort |
| Specific Alternative Flow | Step 8: IF DATA packet TID is invalid THEN | |
| | 1 | Create & send ERR packet (code 5) to the invalid TID |
| | 2 | Continue transfer |